Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Combat resource allocation planning in naval engagements

A. Benaskeur
É. Bossé
DRDC Valcartier

D. Blodgett
Lockheed Martin Canada Inc.

Defence R&D Canada – Valcartier
Technical Report
DRDC Valcartier TR 2005-486
August 2007

Combat resource allocation planning in naval engagements

A. Benaskeur
É. Bossé
Defence R&D Canada – Valcartier

D. Blodgett
Lockheed Martin Canada Inc.

Defence R&D Canada – Valcartier

Technical Report

DRDC Valcartier TR 2005–486

August 2007

Principal Author

A. Benaskeur

Approved by

Éloi Bossé
Head/A Section

Approved for release by

C. Carrier
Chief Scientist

Abstract

Reaction times of modern current and future war platforms are eroded, since they are expected to operate in a large variety of complex scenarios. To cope with the increasingly diverse air and surface threats, modern platforms, either operating in a single ship configuration or within a task group, will require their sensor suite and weapon arsenal to be efficiently managed. The coordination and tight integration of these resources will also be required.

The Decision Support Systems (DSS) Section, at Defence Research & Development Canada – Valcartier (DRDC Valcartier), has initiated collaboration with industry and university partners. This collaboration aims at developing and demonstrating advanced concepts of combat resource management. The latter could apply to the current Command & Control Systems (CCSs) of the Halifax and Iroquois Class ships, as well as their possible future upgrade (*i.e.*, SCSC platform), in order to improve their performance against predicted future threats. This activity builds upon and broadens the scope of prior research in the domain. It is oriented to the study, development, and implementation of management decision aids for tactical shipboard resources, based on intelligent agent technology and techniques for multi-agent planning and coordination.

This report presents a review of agent and multi-agent planning approaches. Theoretical basis of agent and multi-agent systems are introduced and planning problems are described. The results of the implementation and test of different algorithms for hardkill and softkill combat resource allocation, in naval engagements, are presented and discussed.

Résumé

Étant donné la complexité et la grande diversité des scénarios dans lesquels les plates-formes militaires modernes doivent évoluer, leur temps de réaction se voit rétrécir continuellement. Ainsi, pour faire face aux menaces aériennes et de surface de plus en plus diverses, ces plates-formes doivent absolument compter sur une gestion efficace de leur arsenal d'armes et suite de capteurs. La coordination et l'intégration de ces ressources sont également requises.

La Section des systèmes d'aide à la décision (SAD) de Recherche et développement pour la défense Canada – Valcartier (RDDC Valcartier) a entrepris une collaboration avec des partenaires de l'industrie et du milieu universitaire. Cette collaboration a comme principal objectif le développement et la démonstration des concepts avancés de la gestion de ressources afin d'améliorer l'efficacité défensive des plates-formes face aux menaces prévisibles. Ces concepts pourraient s'appliquer aux systèmes de Commandement et Contrôle (C^2) actuels des navires canadiens des classes Halifax et Iroquois, ainsi qu'à leurs versions futures. Cette collaboration poursuit et élargit la portée de travaux antérieurs dans le domaine. Elle vise à étudier, développer et implanter des outils d'aide à la décision pour la problématique de la gestion des ressources tactiques embarquées. Ce travail se base sur la technologie des agents intelligents et les techniques de coordination multi-agents.

Ce rapport présente une revue de différentes approches de planification pour les systèmes d'agents et multi-agents dont les fondements théoriques et les problèmes de planification sont exposés. Les résultats de l'implantation et de l'expérimentation des différents algorithmes pour l'allocation des ressources de destruction ('hardkill') ou de mise hors combat ('softkill'), dans des engagements navals, sont présentés et discutés.

Executive summary

Combat resource allocation planning in naval engagements

A. Benaskeur, É. Bossé, D. Blodgett; DRDC Valcartier TR 2005–486; Defence R&D Canada – Valcartier; August 2007.

Management of tactical combat resources, as a part of military naval Command & Control (C^2) process, provides a real world multi-agent application where the agents are both human and software decision-makers. A typical warship, such as a Halifax Class Frigate, uses different modules that interact together to defend herself. It is necessary to propose ways to optimize the allocation and the coordination of the different resources and agents in order to increase the ship defensive effectiveness against threats.

In the recent years, agent and multi-agent technology has become more and more important in many fields. The simple agent technology aims at conceiving entities capable of acting in a rational way. However, in many applications, the agent alone is insufficient to do all the tasks, and it is preferable to view it evolving with other agents. This leads to conceive a multi-agent system, where agents interact together in order to plan, cooperate, compete, or more simply coexist.

One specific interest in the reported work turns around the use of multi-agent planning. Planning with Multi-Agent Systems (MAS) is explored with, as a main goal, the conception of a combat Resource Management System (RMS) for a generic military ship. That RMS could ultimately be extended to the Halifax and Iroquois Class ships, as well as to the future Single Class Surface Combattant (SCSC) platform. The report presents some theoretical bases on real-time planning and resource allocation. Different approaches to handle contingencies are also discussed. Three planning approaches were implemented and tested. These include: Partly Planner, Holistic Re-engagement Planner, and Holistic Tabu Planner, for hardkill allocation planning. Simple rules (purely reactive planner) are used for the deployment of softkill weapons. These algorithms were tested and their respective advantages and disadvantages are discussed. A test bed environment was developed to investigate the different algorithms for Anti-Air Warfare (AAW) hardkill and softkill allocation planning. The developed Naval Display Simulator uses JACK^{TM}Agent development tool.

This page intentionally left blank.

Sommaire

Combat resource allocation planning in naval engagements

A. Benaskeur, É. Bossé, D. Blodgett; DRDC Valcartier TR 2005–486; Recherche et développement pour la défence Canada – Valcartier; août 2007.

La gestion des ressources tactiques fait partie du processus militaire de Commandement et Contrôle (C^2) naval. Elle fournit une application à caractère multi-agent. Les agents y sont des décideurs humains ou des agents logiciels. Une frégate de classe Halifax utilise différents modules qui interagissent ensemble. Afin d'améliorer l'efficacité des capacités défensives de la frégate face aux menaces, une optimisation de l'allocation et de la coordination de ses différentes ressources et des agents est requise.

La technologie des systèmes d'agents et multi-agents est de plus en plus présente dans beaucoup de domaines. La technologie agent permet de concevoir des entités capables d'agir rationnellement. Toutefois, dans beaucoup d'applications, un agent seul est incapable de réaliser toutes les tâches requises. Il est alors préférable de le voir évoluer avec d'autres agents, ce qui a amené à concevoir un système multi-agent, dans lequel l'interaction entre les agents permet la planification, la coopération, la compétition ou, tout simplement, leur coexistence.

Un aspect du travail rapporté dans ce document trait tout particulièrement à la planification dans un contexte multi-agent. Celle-ci est explorée avec comme objectif principal la conception d'un système de gestion des ressources pour une frégate générique. Les résultats de ce travail pourraient par la suite être adaptés, tant aux navires des classes Halifax ou Iroquois, qu'à la future plate-forme de combat canadienne. Dans ce rapport, la base théorique des systèmes d'agents et multi-agents est présentée et les problèmes de planification en temps réel et de l'allocation des ressources sont abordés. Différentes approches, pour traiter les contingences, sont également discutées. Trois approches de planification ont été implantées et testées. Il s'agit de 'Partly Planner', 'Holistic Re-engagement Planner', et 'Holistic Tabu Planner', pour la planification de l'allocation des ressources de destruction ('hardkill'). Des règles simples (un planificateur réactif) ont été utilisées pour le déploiement des ressources des mise hors de combat ('softkill'). Ces algorithmes ont été testés et leurs avantages et inconvénients respectifs sont discutés. Un banc d'essai, baptisé Naval Display Simulator, a été développé. Ce dernier utilise l'outil de développement JACK$^{\mathrm{TM}}$Agent et permet l'expérimentation des différents algorithmes.

This page intentionally left blank.

Table of contents

This page intentionally left blank.

List of figures

This page intentionally left blank.

List of tables

This page intentionally left blank.

Acronyms

AAW	Anti Air Warfare
ACL	Agent Communication Language
AI	Artificial Intelligence
ASM	Anti-Ship Missile
ASuW	Anti-Surface Warfare
ASW	Anti-Submarine Warfare
AWW	Above Water Warfare
BDI	Belief, Desire, Intention
C^2	Command & Control
CCS	Command & Control System
CIWS	Close-In Weapon System
CORBA	Common Object Request Broker Architecture
CPA	Closest Point of Approach
CPU	Central Processor Unit
CWI	Continuous Wave Illuminator
DAI	Distributed Artificial Intelligence
DAMAS	Dialogue, Agents, Multi-AgentS
DND	Department of National Defence
DPS	Distributed Problem Solving
DRDC	Defence Research & Development Canada
DSS	Decision Support System
EA	Electronic Attack
ECM	Electronic Counter Measure
EM	Electromagnetic
ESM	Electronic Support Measures
FCR	Fire Control Radar
GMVLS	Guided Missile Vertical Launch System
GUI	Graphical User Interface
HCI	Human-Computer Interface
HVU	High Value Unit
ICL	Inter-Agent Communication Language
IDL	Interface Definition Language
IFF	Identification Friend or Foe
IR	InfraRed
IRST	InfraRed Search & Tracking
JVM	Java Virtual Machine
KA	Kill Assessment
KQML	Knowledge Query and Manipulation Language
LM Canada	Lockheed Martin Canada
MAS	Multi-Agent System
MCG	Medium Caliber Gun
NDS	Naval Defence Simulator
NSERC	Natural Sciences and Engineering Research Council

OAA	Open Agent Architecture
PGP	Partial Global Planning
PK	Probability of Kill
PP	Partly Planner
RAMSES	Reprogrammable Advanced Multi-mode Shipboard Electronic Counter Measure System
RCS	Radar Cross-Section
RCSR	Radar Cross-Section Reduction
RDDC	Defence Research & Development Canada
RF	Radio Frequency
RM	Resource Management
RMS	Resource Management System
ROE	Rules Of Engagement
SAM	Surface-to-Air Missile
SCSC	Single Class Surface Combatant
STIR	Separate Tracking and Illuminating Radar
TOF	Time Of Flight
UML	Unified Modeling Language
VLSS	Vertically Launched Sea Sparrow
VOI	Volume of Interest
WTA	Weapon-Target Allocation

Acknowledgments

We would like to thank several people, from Dialogue, Agents, Multi-AgentS (DAMAS) Group at Université Laval, for their active participation in this project: Pierrick Plamondon, Sébastien Paquet, Matin Soucy, Patrick Beaumont, Jean-François Morissette, and Brahim Chaïb-Draa. A special thanks goes to Yaïves Ferland, from DRDC Valcartier, for his in-depth reviewing of this report and his valuable remarks and corrections.

This page intentionally left blank.

1 Introduction

Advances in threat technology, the increasing difficulty and diversity of open-ocean and littoral (*i.e.*, near land) scenarios, and the volume and imperfect nature of data to be processed under time-critical conditions pose significant challenges for future shipboard Command & Control Systems (CCSs). Among other functionalities, a CCS provides operators with capabilities to evaluate the threat level of the different objects that are present within the Volume of Interest (VOI). When deemed necessary, the CCS uses the shipboard combat resources to respond to those threats. However, current operational systems generally provide little support for tactical decision making in complex, highly dynamic scenarios where time for decision making and action execution is at a premium. The need for such support is all the more pressing given the current emphasis on littoral warfare, where reduced reaction times and complex Rules Of Engagement (ROE) are the norm.

Management of tactical shipboard combat resources, as part of military naval Command & Control (C^2) process, provides a real world application that involves both human and software decision makers. To defend itself, a naval platform, such as a Halifax Class Frigate, uses different systems and modules that interact directly or indirectly together. Therefore, it is very necessary to propose ways to allocate and coordinate the use of the different systems in order to increase the ship's defensive effectiveness against potential threats.

Defence Research & Development Canada – Valcartier (DRDC Valcartier), with its partners from Canadian universities and industry, have for several years now been investigating methods to augment or enhance existing shipboard CCS capabilities. DRDC Valcartier was involved, among others, in a collaborative grant with NSERC, Lockheed Martin Canada, Université Laval, and Université de Montréal to investigate, design, develop and implement a real-time Decision Support System (DSS). The latter can be integrated into a ship's CCS to assist operators in conducting the tactical Command and Control (C^2) process, focusing on naval combat Resource Management (RM) in the context of Above Water Warfare (AWW). This collaboration built upon and broadened the scope of prior work [1]. It aims at exploring concepts concerned with multi-agent techniques for the design, development, implementation, and evaluation of a computer-based, real-time DSS to assist operators in conducting tactical C^2 process, with an emphasis on combat RM.

To achieve the above stated primary goal, the following objectives were defined for the reported project:

1. To review and evaluate real-time planning and coordination mechanisms (multi-agent planning & scheduling) for application to AWW combat RM problem.
2. To specify, develop, and validate planning and coordination techniques to enable one or more platforms in order to defend themselves in an efficient way against incoming threats. Coordination concerns both single-ship hardkill/softkill coordination and multi-ship plan coordination [2].
3. To consider the ship positioning as a resource that should be managed like any other resource.

4. To develop a simulation and testing capability and review its efficiency according to the software engineering approach.

Finally, the ultimate objective is the contribution to the development of methodological knowledge and skills so much needed in Canada to meet the challenge of decision making in the context of military RM. This effort will allow the Department of National Defence (DND), the universities and industry partners to acquire knowledge and expertise in this domain.

Most of the work presented in this report was achieved under the above-mentioned collaboration, whose one of the major objectives is to explore concepts concerned with *agent* and *multi-agent* planning and coordination technology, with application to the tactical naval combat RM for a frigate-like platform. The focus on *agent* and *multi-agent* technology was motivated by the fact that, in the recent years, this technology has become more and more important in many fields such as computer engineering, industrial engineering, etc. Other reasons that sustain the choice for the multi-agent techniques are that tactical combat RM is a complex process, is a distributed application, and needs coordination and cooperation between different entities in order to manage its resources.

Consequently, the multi-agent technology was adopted throughout the reported collaboration to address different problems. This choice is motivated by both the need to address a real world problem with concepts from the Distributed Artificial Intelligence (DAI) and Multi-Agent Systems (MAS), and the need to find an efficient coordination method between the different entities participating in the tactical combat RM process. The work and results of this research effort are described in a series of reports. The current one concerns the naval combat resource allocation planning problem and is organized as follows.

1.1 Organization of the report

Naval combat RM problem is first introduced in Chapter 2, where some generic issues relative to military C^2 systems are also discussed. In Chapter 3, agent and multi-agent systems are described. Contingency is introduced and some different categories to solve contingencies are established. Chapter 4 describes the investigation of Anti-Air Warfare (AAW) hardkill and softkill planning. Chapter 5 provides report and project conclusions, including recommendations for future work. The specifications for the ownship resources used in this project are described in detail in Appendix A. It is very important to understand all the resources and how they work because they inspire and bound the concepts investigated in this project. Because these resources are representative of the type found aboard Halifax (and, to some extend, Iroquois) Class warships, the results of this project will also be generally applicable to those platforms. Appendix B presents the simulator used to test the different defence planning strategies. This simulator has been implemented using Java and JACK$^{\text{TM}}$programming languages. The rationale for selecting JACK$^{\text{TM}}$is discussed in Appendix B, as well.

2 Naval combat resource management

There are some generic issues in military C^2 Systems (CCSs) that indicate that the development of a relevant C^2 theoretical frame will have significant impact upon the analysis and design of both military and civil CCSs. In particular, the focus here is on issues related to organizational forms and distributed decision architectures, which are precisely the areas that offer the most fertile ground for both basic and applied research.

1. **C^2 is a distributed environment** – Solving the C^2 related problems involves both human and software decision makers. The latter may be geographically dispersed due to the operational environment, the nature and characteristics of the threat and/or the configuration of the ownship itself. All those contribute to the distributed architecture of CCSs. Cooperation, coordination, and communication between the decision makers are thus crucial in such a distributed architecture. The military CCS is often modeled as a multi-agent organization, in which the decision agents are both human and software decision makers.

2. **C^2 has a functional architecture** – Another key element of the C^2 process is its functional decomposition into a set of generally accepted C^2 functions that must be executed in some reasonable delays to ensure mission success. A list that gives a very high-level description of those functions, related to defensive battle management problem, is given below.
   (a) **Target detection** - It depends on the performance of sensors , and may be based on data from a single sensor or a combination of several sensors.
   (b) **Target tracking** - It uses the sensor data to optimally estimate the current kinematical properties of the targets, and predict their future positions. It is generally based on data fusion techniques.
   (c) **Target identification/classification** - The identity and class information of targets are established. This also results in the resolution of true targets from decoys or non-hostile objects.
   (d) **Threat Evaluation** - It establishes the intent and the capability of the potential threat within the Volume Of Interest (VOI).
   (e) **Weapons Assignment** - In this process, decisions are made on how to deal with the identified threats. This process can be subdivided into several sub-problems that include mainly the following three ones:
      i. **Response Planning** - This includes the combat resource *allocation*[1] and the combat resource *coordination/cooperation*[2]. During the combat resource allocation, one or more weapons are assigned to engage each threat, including the assignment of supporting resources (as sensors, communications, etc.) required for each and every one-to-one engagement. Combat resource

---

[1] Which is the core problem addressed in the remaining of this report.

[2] The problem of combat resource coordination/cooperation is outside the scope of this report and is treated in detail in [2].

coordination is about conflict (negative interaction) resolution, while combat resource cooperation is about synergy (positive interaction) exploitation.

   ii. **Response Execution** - The process by which the planned response is executed in real-time.

   iii. **Outcome Assessment** - The process by which the outcome (ownship damage or threat damage/kill) of the executed actions is evaluated.

   This process necessitates a highly dynamic flow of information and decision making that involves a number of operators and sophisticated support capabilities.

3. **C^2 is a complex process** – The AWW problem is a very complex problem, and this complexity often rises from the multitude, the heterogeneity and the inter-relationships of the resources involved. This is in general the case when simultaneous engagements involving heterogeneous sensor and/or weapon systems can take place, and human commanders make a large part of the decisions. Generally, no commander alone can deal with the inherent complexity of the entire engagement; this leads to a decomposition of the decision process along distinct expertise or know-how dimensions. In the light of these considerations, team training is essential, in a C^2 organization, so as to achieve superior coordination and to make the best utilization of scarce common resources. Moreover, a military CCS must take into account the specific command and decision established hierarchy.

4. **C^2 deals with large volumes of data under stringent time constraints** – Perceptual and cognitive processing is complicated by the fact that the information is derived from a variety of organic and non-organic sources. Those sources include radar, Electronic Support Measures (ESM), Infra-Red Search and Track (IRST), Identification Friend or Foe (IFF) transponder responses, as well as intelligence information from shore and various deployed units. Particular processing problems are caused by the fact that: i) non-organic information is generally less timely than organic information, which makes it difficult to correlate the two types of information; and ii) the data to be integrated are generally imperfect[3]. It follows that operators may have to handle potentially large situation uncertainties and at any given moment there may be several likely interpretations of the tactical picture. This leads to processing large volumes of data under stringent time constraints.

2.1 Combat resource management

As part of the naval C^2, the combat RM problem is subject to a number of properties and constraints, inherent to the very nature of RM problems in dynamic environment, generally, or imposed by the military and naval contexts. The following are few of the most relevant [**?**] properties that should be taken into consideration in developing the management capabilities.

---

[3] It can be uncertain, incomplete, imprecise, inconsistent, and ambiguous, or some combination of these, due to limited sensor coverage, report ambiguities, report conflicts, or inaccuracies in the measured data

1. **Deterministic vs. Stochastic** – If the next state of the environment is completely determined by the current state and the action to be executed by the decision maker or the planner, then the environment is said to be deterministic; otherwise, it is stochastic by nature. The combat RM problem is very stochastic. The manager cannot exactly predict the evolution of the environment according to the current state (for example: emergence of new threats, changes in threat trajectories , own resource performance, etc.).

2. **Episodic vs. Sequential** – In an episodic environment, the decision-making problem is divided into atomic episodes. Each episode consists in perceiving and then performing a single action. Crucially, the action selection process in a given episode does not depend on the actions taken in previous episodes. Therefore, the choice of actions in each episode depends only on the episode itself. In sequential problems, the current decision could affect all future decisions. The combat RM problem is sequential in the sense that making any decision can affect and constraint subsequent decisions (*e.g.*, committing a given resource against a given threat can have an impact on the availability of that resource for subsequent engagements). Sequential problems are much harder than episodic ones, because the decision making process need to think ahead in future about consequences of current decisions and actions. The simplification, sequential problems are often treated as episodic. Under severe time constraints, this may lead to very undesirable results.

3. **Static vs. Dynamic** – If the environment may change during the decision-making process, then the environment is said to be dynamic; otherwise, it is static. Static environments are easy to deal with because the decision-making process does need to worry about time. However, time is a central concern when dealing with dynamic environments. Continuous action is required to cope with changes in the environment. The combat RM problem may take the two forms:

   (a) In the static version of the combat RM problem, all the inputs to the problem are fixed; that is, all targets are known, all weapons are known, and all weapons engage targets in a single stage. This concerns mainly the off-line planning for tactics generation.

   (b) The dynamic version of the combat RM problem is a multi-stage problem where the environment may change very quickly and very often (*e.g.*, threats keep moving, maneuvering, appearing, disappearing, etc.) during the response planning process. Also, in the dynamic version, when some weapons engage the targets at a given stage, the outcome of this engagement is first assessed, and a strategy for the next stage is then decided. This is called a "shoot-look-shoot" strategy since the defence is alternating between shooting the weapons and observing (assessing) the outcomes. In such a context, the reaction time (to the environment changes) becomes the main issue. It is almost always possible to find an optimal response to a given situation. However, the issue remains to provide it on time. This is why, in very dynamic environments, optimal responses are seldom achievable. The enormous combinatorial complexity of the problem implies that, even with the supercomputers available today, optimal solutions cannot be obtained in real-time. Rather, what are sought after are satisfying responses; that is the best ones given the constraints.

4. **Single vs. Multi-Criteria** – If the decisions to be made are evaluated according to a single criterion, the problem is said to be single-criterion, otherwise it is said to be multi-criteria. In the latter case, that represents most of real life problems, decisions are evaluated based on several criteria that may be in conflict with each other. Examples of naval combat RM include threat level, own resources effectiveness, and cost of actions.

In summary, the combat RM problem, that is concerned by this work, is a distributed, stochastic, sequential, dynamic and multi-criteria one. The project ultimate goal is to allocate, coordinate, and schedule the use of the shipboard combat resources over a time horizon that provides for and optimizes single ship/point defence, as a primary objective, and ultimately multiship/area defence capabilities. Note that the ship has, as described in the next section, a set of tactical resources that allows it to defend itself.

2.2 Shipboard combat resources

The exact nature of the specifications and capabilities of the various AAW weapons on the Canadian warships is obviously very complex, and much of that information is classified by the DND. To avoid this issue, and in order to maintain emphasis on the research interests and not be burdened by the complexity and fidelity of the representation, a considerably simplified model of the relevant AAW weapons was used for this project. This model is a simple,unclassified version of AAW weapons for a typical frigate. The results could eventually be applied to the Canadian warships of Halifax Class, to some extend to the Iroquois Class, given that the latter shares to same layered defence configuration with the former. The details of that unclassified model can be found in Appendix A.

2.2.1 Hardkill resources

The AAW hardkill are weapons that are directed to intercept a threat and actively destroy it through direct impact or explosive detonation in the proximity of the threat. The range of different types of hardkill weapons varies, and the effectiveness of these weapons depends on a variety of factors[4]. The AAW hardkill weapons for a typical frigate include Surface-to-Air Missile (SAM) systems that have the greatest range, an intermediate range Gun, and a Close-In Weapon Systems (CIWS) that is a short-range, rapid-fire gun. Closely allied to these weapons are two Separate Tracking and Illuminating Radars (STIRs) that are used to guide a SAM to a threat, and to point the Gun. This effectively provides two concurrent fire channels for the AAW hardkill weapons. The CIWS has its own pointing radar.

---

[4] *E.g.*, the distance to the threat, the type of threat, the speed of the threat, the environment, etc.

2.2.2 Softkill combat resources

The AAW softkill weapons use techniques to deceive or disorient a threat to cause the threat to destroy itself, or at least lose its fix on its intended victim. Again, the range and effectiveness of these weapons varies considerably. The AAW softkill weapons for a typical Canadian frigate include chaff and jamming systems. The chaff system launches a shell that produces a burst at a designated position. The resultant chaff cloud has a significant radar cross-section that can be used to screen the ship or produce an alternate target on which a radar-guided threat can fix. The jamming system uses electromagnetic emissions to confuse the threat's sensors to cause the threat to either lose its fix on its intended target, or to improperly assess the position of its target.

During an attack, jamming and chaff systems must act concurrently and in a complementary way. First, the jammer is used to break the missile threat's radar lock on ownship. Once the missile has lost its target, the jammer creates a false target position on the missile's radar. Then, chaff is deployed at a position consistent with the false one provided by the jammer. In this way, the missile's radar locks onto the chaff cloud as its new target.

Note that, due to their different mechanisms, the hardkill and softkill weapons have historically led independent existences in terms of design and operational deployment. Generally, the hardkill and softkill weapons are supervised by separate control personnel. Thus, the complex task of optimally combining the two weapon types falls squarely on the shoulders of the person responsible for overall air defence. The inherent differences between hardkill and softkill weapons, and the nature of their deployment history on typical warships, lead naturally to a representation of hardkill and softkill as being two software agents, so that each determine a real-time plan for its resources and both have to coordinate plans between them.

2.2.3 Sensors

As a part of the tactical combat resources, the ship has two classes of sensors: surveillance and fire support. For the surveillance, the ship possesses the AN/SPS-49, which is an L-band, long-range, two-dimensional, air-search radar system that provides automatic detection and reporting of targets within its spatial VOI. The AN/SPS-49 is used for early target detection. The SG-150 is the multi-purpose air and surface search naval radar developed by Ericsson.

The Halifax Class Frigate combat resources comprise two Fire Control Radar (FCR) systems called respectively STIR-A and STIR-B. STIR-A is installed on the roof of the bridge and STIR-B on the raised radar platform immediately forward of the helicopter hangar. The STIRs are responsible for the control of the SAM fire channels and the Gun. They provide the SAM and the Gun weapon systems with fire control quality track data for engageability and fire control calculations.

Sensors, for both surveillance and fire support, need to be coordinated with weapons deployment in order that maximize the defence effectiveness.

2.2.4 Ship navigation

The position and the maneuvers of the ship play a key role in its defensive plan. Therefore, the ship navigation is treated as combat resource that needs to be coordinated with the other combat resources deployment in order to increase the ship's survivability. In this report, the focus will be more particularly on the combat resource allocation planning problem. The combat resource coordination/cooperation problem is addressed in [2].

2.3 Resource allocation problem

From the technological viewpoint, a resource is any substance or (set of) object(s) whose cost or available quantity induce some constraint on the operations that require to use it. For instance, hardkill naval engagements require fire control radars. A STIR can track and illuminate only one target at a time. Therefore, for SAM and the Gun, only two fire control channels are available, which put hard constraints on their allocation process in situations that involve more than two planned engagements. The resources may be of different nature, depending on the problem at hand. Nevertheless, most resources belong to one the following categories:

1. **Unary** resources are resources whose capacity is 1. Two activities requiring the same unary resource cannot overlap. Examples of this type of resource are given by the Gun and the CIWS.

2. **Discrete** resources are resources of capacity Q. Activities requiring the same discrete resource can overlap provided the resource capacity Q is not exceeded. Examples are given by the STIRs and the jammers.

3. **Reservoir** resources are resources of capacity Q and initial level L. Activities may consume or produce the reservoir. The level of the reservoir over time must be kept in the interval $[0, Q]$. This may represent the ammunition magazine. This type of resources will not be considered in the remainder of the report.

Resources are tightly linked with the notions of time and concurrency. For unary and discrete resources, activities use the resource over some time interval. Handling parallel execution of activities is necessary to take advantage of non-unitary capacities. There may be complex relations that link the duration of an activity with the amount of resource it uses, consumes and/or produces. Therefore, the combat resource allocation problem boils down to two classes of problems, which are the *resource allocation planning* and the *resource allocation scheduling*. The border between the two is often very fuzzy and very dependent on the targeted problem. The following definitions aim at making this border as crisp as possible for the problem of interest.

2.3.1 Resource allocation planning

Resource allocation is about the assignment of resources to activities, where the start and end times of each activity are given. In dynamic contexts, this represents a continuing process of analyzing relevant information from the present and the past, and then the assessment of probable future developments so that an allocation strategy may be determined, enabling the overall system to meet its stated objectives. In the military context, one often refers to as the Weapon-Target Assignment (WTA) [3]. The problem consists in optimally assigning weapons to the enemy-targets so that the total expected survival value of the targets after all the engagements is minimized. Efficient solutions to this problem are of great interest to the military. The reason for this is that, in an engagement with the enemy, the problem must be solved in real-time. The enormous combinatorial complexity of the problem implies that, even with the supercomputers available today, optimal solutions cannot be obtained in real-time. Good heuristics must therefore be developed to solve the problem [4].

In summary, in the naval RM problems, resource allocation planning consists in selecting which weapons should engage which threat, independently of the order in which the different engagements will actually take place.

2.3.2 Resource allocation scheduling

Resource allocation scheduling consists of assignment start and end times to activities, where each activity requires given resources with given capacities. In pure scheduling problems, activities are already chosen (or given), leaving only the problem of determining a feasible order among them. In naval operations, scheduling determines when a specific defensive action (a specific weapon against (a) specific threat(s)) will take place. Schedule constraints specify when an action should start or end based on duration, predecessors, resource availability, or intended interception time.

It is, however, very important to mention that the allocation of tactical combat resources in naval engagements involves the two above-mentioned classes of problems. It requires:

1. Reasoning about limited time and scarce resources is at the very core of the tactical RM problem, such as in purely scheduling problems.
2. RM problem also involves choices. This problem cannot just be confined to a task ordering, but includes choices to make about which resources to use for each task (action). For a given task, several alternative resources may be available while having different cost and/or durations, such as in pure allocation planning problems.

Therefore, the resource allocation problem is about the allocation of both the resources and start and end times to activities. Therefore, this defines a *joint* resource allocation planning & scheduling problem. Note that, beside the two above mentioned fundamental problems, a set of different problems needs to be addressed within the combat RM context. These problems are either imposed by the nature of the primary problem to be solved, and/or by the environment in which it must be solved, or even by the approaches chosen to solve it.

This page intentionally left blank.

3 Agent and multi-agent systems (MAS)

The agent technology aims at conceiving entities capable of acting in a rational way through approaches turning around planning, uncertainty reasoning, decision theory, machine learning, vision and perception, etc. However, in many applications, the agent alone is insufficient to do all the tasks, and it is preferable to view it evolving with other agents. This justifies considering multi-agent systems. In this kind of systems, the agents interact together in order to cooperate, compete, or more simply coexist. Agent and multi-agent systems represent a new way of analyzing, designing, and implementing complex systems [5]. The agent-based view offers a powerful repertoire of tools, techniques, and metaphors that have the potential of improving the way in which people conceptualize and implement many types of software. Before embarking on further discussion, terms as “agent”, “agent-based system” and “multi-agent system” need to be defined first. Even though there are no universally accepted definitions of such concepts, definitions adapted from [6] will be used.

3.1 Agent

An agent is a computer system situated in some environment and capable of flexible autonomous action in order to meet its design objectives. There are thus three key concepts in the adopted definition: **situatedness**, **autonomy** and **flexibility**.

1. **Situatedness** means, in this context, that the agent is in a position to receive sensory input from its environment and that it can perform actions that change the environment and its relationship with it in some way. Such situatedness may be contrasted with the notion of *disembodied* intelligence that is often found in expert systems.
2. **Autonomy** is used in the sense that the system should be able to act without the direct intervention of humans (or other agents), and that it should have control over its own actions and internal state.
3. **Flexibility** means that the agent is responsive, pro-active and social.
   (a) **responsive** – the agent perceives the environment and responds in a timely fashion to changes that occur within it;
   (b) **pro-active** – the agent does not simply act in response to the environment, it should be able to exhibit opportunistic, goal-directed behavior and take initiative where appropriate;
   (c) **social** – the agent is able to interact, when appropriate, with other agents (artificial and/or human) in order to complete its own problem solving and to help others with their activities.

The presence of all the previous attributes in a single software entity provides the power of the agent paradigm and distinguishes agent systems from related software paradigms, such as object-oriented systems, and expert systems.

Current interest in agents did not emerge from a vacuum. Researchers and developers from many different disciplines have been talking about closely related issues for some time.

The main contributors to agents are Artificial Intelligence (AI) [7], object-oriented programming [8] and concurrent object-based systems [9, 10], and Human-Computer Interface (HCI) design [11]. Undoubtedly, the main contributor to the field of agents is AI. Ultimately, AI is all about building intelligent artifacts, and if these artifacts sense and act in some environment, then they can be considered as agents [7]. AI planning research is the sub-field of AI that concerns itself with knowing *what to do*, *i.e.*, what action to perform. Ultimately, an agent is just a system that performs actions in some environment, and so it is not surprising that AI planning research should be closely involved in the study of agents (see Section 3.3) .

3.2 Multi-agent systems (MAS)

Traditionally, research into systems composed of multiple agents was carried out under the banner of Distributed Artificial Intelligence (DAI) and has historically been divided into two main camps [12]: Distributed Problem Solving (DPS) and Multi-Agent Systems (MAS). More recently, the term "multi-agent systems" has come to have a more general meaning and is now used to refer to all types of systems composed of multiple autonomous components. DPS considers how a particular problem can be solved by a number of modules (nodes) that cooperate in dividing and sharing knowledge about the problem and its evolving solution. In pure DPS, all interaction strategies are incorporated as an integral part of the system. In contrast, research in MAS is concerned with the behavior of a collection of possibly pre-existing agents aiming at solving a given problem. A MAS can be defined as a loosely coupled network of nodes (agents) that work together to solve problems that are beyond the individual capabilities or knowledge of each node [13]. These problem solvers are autonomous and may be heterogeneous in nature. Multi-agent systems are ideally suited to represent problem solving, but have the additional advantage of offering a sophisticated pattern of interaction. Examples of common types of interactions include: working together toward a common aim (*i.e.*, **cooperation**), organizing the problem solving activity so that harmful interactions are avoided and beneficial interactions are exploited (*i.e.*, **coordination**), and coming to an agreement which is acceptable to all the parties involved (*i.e.*, **negotiation**). It is the flexibility and high-level nature of these interactions that distinguish multi-agent systems from other forms of software. They also provide the underlying power of the paradigm. Cooperation, coordination, and negotiation problems, in the context of combat RM, are treated in [2].

3.3 Agent-based planning

This section presents and discusses the practical agent-based planning problem. The fundamental issue of when to *Deliberate* and when to *React* in agent-based planning systems is discussed. *Deliberation* is used in the sense of design of a sequence of steps to carry out a particular task or achieve a particular goal *before* execution. This is often simply referred to as *planning*. *Reaction* is defined as a set of predefined rules to accomplish a certain goal. Reactive plan is therefore defined as a set of tests and reactions able to solve any one among a set of predicted contingencies.

The real world places an additional important constraint on the planning and execution processes: the resources of real agents are inherently limited. This includes both computational resources as well as tactical physical resources[5]. This means that the agent cannot prepare a response that includes a complete conditional branch to every possible contingency (see Subsection 3.3.3 for definition) that may appear during its plan execution (thus ruling out universal plans as a feasible alternative). Conditional branches are parts of the plan whose execution is conditional to the outcome of a logical test.

In most interesting real-world domains, the number of contingencies that may conceivably appear during the execution of a nontrivial plan is infinite. It is therefore likely that in most useful domains, an agent might:

1. prepare complete conditional branches for only a few contingencies (if any);
2. prepare reactions for a larger, but still relatively small number of "important" contingencies; and then
3. leave the other contingencies untreated at planning time. They either will be ignored at execution time, or the agent will be capable of replanning a solution when such a contingency is detected during the plan execution.

3.3.1 Deliberation

Deliberation consists in designing a sequence of steps to carry out a particular task or achieve a particular goal before execution. Among the strengths of the planning is the fact that plans can be built to have a set of desirable global properties regarding the goals to be attained and the resources available. The side effects of the actions to be executed as parts of the plan can be carefully taken into account and analyzed before execution begins. These properties are achieved by taking into account complete descriptions of the states of the world as the planner predicts them. Of course, these states may conform to reality only if the environment behaves according to the model that the planner has about it. The more incomplete this model is, the more uncertainty in the behavior of the environment, and the more uncertainty about the actual states that will be encountered during the plan execution. However, the planning has always some weakness with respect to the real world. In fact, its two main disadvantages are:

1. **High computational cost of planning** - this makes it necessary to carefully consider which contingencies should be treated exhaustively. Otherwise, it may require an excessive amount of time to build the plan.
2. **Lack of flexibility of the planned behavior** - the system can only act in states of the world that are specified in the plan. Its performance will deteriorate very abruptly with any variations to such states.

---

[5] *E.g.*, the amount of SAMs available for the ownship at any point during the plan execution.

3.3.2 Reaction

A reaction-based planning system consists of reflex planner (possibly with internal state) that can be implemented with any variety of representations of condition-action rules. Therefore reactive planners react to the situation in the environment, without reasoning about it. This consists in predefining a set of rules to accomplish a certain goal. A reactive plan represents a set of tests and reactions able to solve any one among a set of predicted contingencies. However, there are situations where many actions can “fire” simultaneously. There must obviously be a mechanism to choose between the different fired actions. Each response is less carefully analyzed than in the deliberation-based case. This because the former it is only required to stabilize the situation and does not need to embody a complete solution to the final goal. This will provide some more time for replanning.

The main advantage of using a reactive approach is its lower complexity compared with more sophisticated solutions. Nevertheless, there are several fundamental inherent limitations. Since the planner does not use any model of the environment, it will not have enough knowledge for selecting the most appropriate actions. Also, it is difficult to see how purely reactive planners can be designed to learn from experience, and then improve their performance over time. Ultimately, there is no principled methodology for building reactive planners. One must use a laborious process of experimentation, trial, and error.

Since deliberating and reacting complement each other, one should envision a system that first plans courses of action designed to achieve goals under certain anticipated contingencies. Conditional branches are built in the plan for the very likely contingencies that also require significant planning to reach its goal. Then plans are then augmented with context-dependent reactions for noticing and responding to less likely but important exogenous events. The plans execution is monitoring for, and when appropriate, reactions associated with particular phases of the plans are executed. Finally, the plans are revised if local reactions do not adequately address unanticipated events.

3.3.3 Contingencies

Contingencies define any state of the world entered by the system while executing a plan, which is not 1) an exogenously generated state of the world assumed in the design of the plan; or 2) a direct consequence of the actions foreseen within the plan up to that point in time. The term contingency is also used to mean any event, fact, or sign that was not expected as a result of the plan execution, and that triggers a (desired or undesired) change in the state of the world, not expected at that time in the plan. Contingencies are the effects of interactions between the agent and the environment. They occur because of:

1. predictable actions of the environment;
2. the unpredictability of the environment; or
3. the unpredictability of the agent’s execution subsystem.

In the real world, the number and variety of contingencies that can occur during the execution of a plan are unbounded. An ideal planner should take into account all these contingencies and build a "universal" plan. This alternative has already been shown to be not feasible for interesting application domains, due to the practical limitations.

Some of the contingencies may have a very high likelihood of occurrence, while also requiring elaborate sub-plans to treat them. Other significantly less likely contingencies may allow for a very short time of response, while having disastrous consequences if the response does not occur in time. Such contingencies probably should be treated reactively. These reactions need not lead the agent to the final goal of the initial plan; it is enough if they can stabilize the situation, avoid the consequences of the contingency, and allow the agent to re-plan a comprehensive solution from the current situation to the final goal. Thus, one needs both *deliberation* and *reaction* to response efficiently to highly changing and unpredictable situations. As discussed previously, each of these two modes has advantages and drawbacks depending on the situations at hand.

3.3.4 Example of contingencies

Let us first consider an example in the context of naval tactical combat RM. Suppose a commander agent has to response to two threats $\{T_1, T_2\}$ using a set of weapons $\{A, B, C, D, E, F\}$ that are planned along two conditional branches, as shown in Figure 1.

$$\begin{tikzcd}
 & & C \arrow[r] & D \arrow[r, Rightarrow] & \boldsymbol{T}_1 \\
A \arrow[r] & B \arrow[ur] \arrow[dr] & & & \\
 & & E \arrow[r] & F \arrow[r, Rightarrow] & \boldsymbol{T}_2
\end{tikzcd}$$

Figure 1: *A conditional plan*

Weapons $\{A, B, C, D\}$ are used against T_1 and $\{E, F\}$ against T_2. In order to achieve its goal, the agent follows a contingency plan already prepared in advance. To begin with, only a small number of contingencies that may appear during the execution of this plan is considered. For one, suppose that because of time constraints, a simple reactive ('Partly') plan must be used instead of a complex conditional branch to face some threat(s). For another, suppose that there is no weapon left (for example, no SAM left for the ownship). The ownship must respond to these contingencies fast enough to avoid a catastrophe. Then it may continue to execute the original defence plan. Alternatively, it may need to re-plan its defence plan if a new threat is detected (T_2) to include it in a new global plan. Also, contingencies can appear at the Kill Assessment (KA) of a certain threat by a weapon. For example, if weapon A succeed to destroy threat T_1, the use of weapon D for the same threat would not be required because it should be useless and squandering.

Consider the following situations from the previous paragraph:

1. KA of a threat after engagement by a certain weapon;

2. new threat is detected by the ownship surveillance system;
3. use of a reactive plan to defend the ownship; and
4. no SAM left.

According to the previous definition, they are all contingencies. They represent any state of the world, entered by the executing agent while following a plan, that is not a direct consequence of executing the actions of the plan up to that point, or is an exogenously generated state of the world that was not assumed in the design of the plan.

Note that a contingency does not necessarily affect the agent or the plan execution. Even when a contingency affects the plan, it does not necessary negatively. For example, a contingency may be a state, which should have been reached along the way, after executing some additional steps of the plan. The agent may detect it and use it to skip the unnecessary steps in the plan.

There are three types of contingencies that can be applicable to the combat RM problem example.

1. **Contingencies for which the planner builds, in the main deliberative plan, complete conditional branches from the contingency state to the global state.** For example, the contingency generated by the KA of a threat T_1. A defensive weapon does not destroy an incoming threat in every instance. Moreover, not responding to it in an adequate manner can have dire consequences[6]. The treatment of this contingency using, for example, another weapon against the threat T_1 also needs an elaborate plan. The latter should be prepared in advance in order to avoid wasting time. Therefore, the planner should design a complete conditional plan to prepare a complete response to this contingency by firing another weapon to the threat T_1 if necessary and possible.
2. **Contingencies for which the agent prepares a reactive response**. These may be combined into reactive plans by a reactive planner, and then attached to appropriate segments of the complete plan provided by the conditional planner. At execution time, some of these contingencies may require extensive replanning after the situation is stabilized. The reaction itself may also suffice to bring the agent's execution back on its initial plan path. The building of a 'Partly Plan' by the ownship resource manager is an example of a response to such a contingency. Thus, it is a very dangerous situation as it may cause the ownship to be destroyed by a threat. However, to counter both T_1 and T_2, the agent can use simple reactive rules[7]. There is no obligation to have a full conditional plan branch to respond to this contingency. If the time pressure is high, planning a simple reaction associated with the detection of this contingency (through monitoring) will be sufficient.

---

[6] The ownship may not respond to the incoming threat with another weapon.

[7] *E.g.*, a SAM and the Gun to T_1 and a SAM to T_2 threat and the CIWS to T_1 and T_2.

3. **Contingencies ignored by the agent at planning time**. Such contingencies are ignored either because their treatment can be left for dynamic replanning[8] or because they are considered less important than the contingencies of the previous types[9]. The treatment at execution time can fall into two following subtypes.
    (a) **Dynamic replanning** - One assumes that the agent will have enough resources at execution time to perform the replanning task. For example, the agent detects during an engagement a new incoming threat. The commander has no alternative defence plan prepared because it is considered a very unusual circumstance, but usually has enough time during the engagement to re-plan for a complete new defence plan to include that new threat.
    (b) **No operation (Noop)** – That is "take no action", either because the consequences of the contingencies are not high enough to warrant an action (*e.g.*, regarding the magnetic compass malfunction, since the heading indicator can be used as a backup for the magnetic compass), or because the agent simply does not have the resources to take an action to solve them (*e.g.*, they have a too short response time allowed). An example would be when the ownship is out of SAMs.

The justification for the classification of the contingencies in three types is mainly related to the limited resources that a planning agent can use. For a few contingencies, the agent can generate complete plans and combine them into a conditional plan. However, the agent's limited planning and execution resources do not allow for too many contingencies to be taken into consideration. Still, at planning time, the agent can prepare reactive responses for a larger set of contingencies. These responses will not ensure full solutions to the global state, but they will give the agent the possibility of dynamically replanning its actions at execution time. But, in no case, can a real agent with limited resources prepare for all possible contingencies in a real-world application domain. Many of these contingencies must be ignored at planning time.

3.3.5 Representation of contingency space

The simplest representation for the "space" of contingencies is a linear space in which contingencies are ordered by their reaction value or criticality, or by their importance. The intuitive notion of reaction value, which should be understood as the value of executing a certain reaction in a response to a contingency, will be used. Alternatively, it can be seen as the value of preparing in advance a reaction to a given contingency. For example, the value of preparing a fast reaction while doing a "Partly Plan" (Chapter 4) for addressing incoming threats is higher than the value of preparing a reaction to a speed indicator malfunction. Indeed, if the former contingency is not solved as soon as possible, the consequences may be fatal. On the other hand, not responding to the latter contingency may not be catastrophic[10]. Figure 2 presents a possible linear representation of the space of contingencies.

---

[8] When they are encountered at execution time.

[9] The agent does not have the resources to prepare a reaction (and much less to prepare a complete branch in the plan) for them.

[10] Since not knowing the speed of the ownship is not so critical to its survival.

It shows that conditional planning is the most critical contingency to be taken into account by a planner. So it has to be planned in advance.

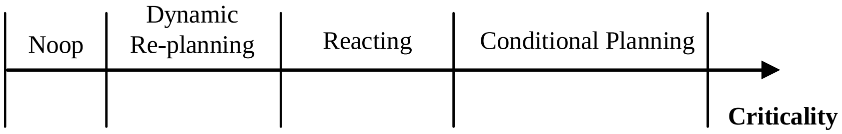


Figure 2: *Linear representation of the contingency space*

In case of the naval combat RM application, the "KA of a threat against after engagement by a certain weapon" corresponds to the conditional planning process and "a new threat is detected by the ownship surveillance system" corresponds to the replanning process. Questions may be asked about why the last contingency is less critical since both contingencies involve the same problem for the ownship, which is to be destroyed by an incoming threat. The reason is that a planner cannot represent all possible contingencies and will consider only the most frequent one. It is assumed that a KA happens more often than the appearance of new threats.

The main disadvantage of this simple linear representation of the contingency space is that it cannot represent other still possible bordering between types. For example, it does not represent the border between *reacting* and *noop*. This may be described by the allowed response time to a contingency: there is no reason in preparing reactions to contingencies that allow for too little response time[11], but there is a good payoff to preparing a reaction to a contingency like the detection of threat(s). Such a contingency needs a fast plan (Partly planning) to counter it, which allows a short time to respond, but long enough so that the response can be taken effectively if prepared in advance.

Another missing common border is between *conditional planning* and dynamic replanning. If the contingency allows for enough time to dynamically re-plan at execution time, then it is better to leave it for that time. When possible, dynamic replanning offers two advantages:

1. At planning time, it saves planning unnecessary conditional branches before the execution starts. This saves planning resources that can be used more productively to prepare for other contingencies.
2. At execution (replanning) time, the agent will have more accurate information regarding the situation and the environment, and is more likely to yield a more accurate plan.

Therefore, a more appropriate representation for the space of contingencies is required. Figure 3 shows a planar representation of the information contained in Figure 2. The "Planning" and "Reacting" values define the space of the contingencies instead of the criticality value as previously stated. Planning value is related to quality of the response to the contingency, while reaction value is related to the timeliness of this response.

---

[11] Like a meteor falling onto the ownship.

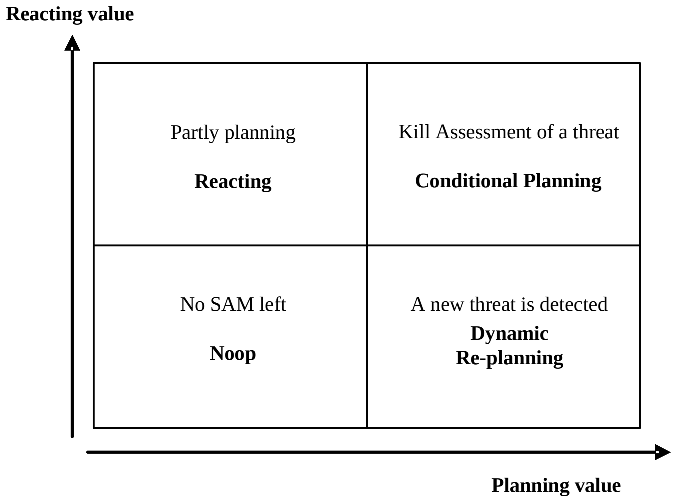


Figure 3: *Planar representation of the contingency space*

There are two qualitative differences between conditional branches and reactions.

1. Conditional plan branches represent global solutions to the initial problem. That is, they are sequences of actions that ensure the agent reaches the goal in the absence of other contingencies. Reactions, on the other hand, are only single (or short sequence of) actions, intended only to stabilize the situation so that the agent can then take its time to re-plan a solution from the state reached after reacting to the initial goal. Therefore, reactions can be seen as the first steps of incomplete conditional plan branches, but at the same time they are more generally applicable than specific branches. There is also no guarantee that after executing a reaction, the agent will find a plan to get it to the initial goal; it is possible that the planner may subsequently find no solution after completing the reaction to the goal. This is not the case for conditional branches, assuming that no other contingencies are encountered. Therefore, it is always assumed that a conditional planned branch is a better solution than a reaction to the same contingency.

2. In conditional planning, the planner has to work out a solution (sequence of actions) from a given state (the contingency) to the goal. In reaction planning, the agent already knows (in its knowledge base) the best reactions, associated with contingencies for applicable classes of situations. The only task of the reaction planner is to combine the reactions associated with the set of contingencies prepared for, into a structure (decision trees, decision list, etc.) that can be conveniently searched at execution time to determine the actual contingency encountered, and its associated reaction. Therefore, planning time is definitely of importance in conditional planning, but may not be an issue when structuring a reactive plan from a set of known reactions. If the planning time cannot be ignored, then the complexity of the reactive plan-structuring algorithm must be taken into account to reduce the set of contingencies for which reactions should be prepared.

3.3.6 Criticality of contingencies

The *criticality* of a contingency is a numerical measure of the amount of damage that can potentially occur to the agent or to the execution of its plan, due to the appearance of that contingency in a certain situation. Therefore, the most important problem here is to decide which contingencies, in a certain situation, are critical enough to require the planning agent to prepare in advance reactive responses or conditional branches for them, and which should be ignored at planning time. In order to be able to define the value of reaction to a contingency and to be able to order the contingencies according to this value, one should identify the characteristics of contingencies that influence this reaction value. Four of those characteristics are considered.

1. **Likelihood of appearance of the contingency in a situation**. For example, for a ship, it is more frequent to have an incoming Anti-Ship Missile (ASM) at 10 km from a coast than an ASM at 1000 km.
2. **Time pressure exerted by the contingency on the agent**. For example, taking care of making a plan for an incoming threat has a highest time pressure than if the radio communication volume becomes very high suddenly, because a plan must be made in a more limited time since it is more critical for the ownship survival.
3. **Gravity of consequences presented by the contingency if no action is taken within the allowed response time**. For example, the consequences are more dramatic when not planning a response to the threat than when not knowing the speed of the ownship.
4. **The side-effects incurred by the reaction to the contingency**. The side effect of replanning the plan to insert a new threat is greater than adjusting the radio volume since the final result for the replanning process can deteriorate the original plan by suppressing engagements to threat(s) in the previous plan. Adjusting the radio volume has no side effect.

The ranking of these four characteristics by each contingency can be obtained from experts in the domain, or through some automatic learning methods.

When all the possible contingencies are ranked, one should decide, in a second phase, which of these contingencies will actually be included in the reactive plan, by taking into account the characteristics of the reactive planner and the limitations on the agent's resources. While making its plan, if the agent reaches a contingency that cannot be responded to within the allowed time period, while still being able to respond to all the contingencies included before it, then this contingency will be left out. This process continues until all contingencies have been examined, since some contingency further down the list may allow a longer response time, while still allowing time to respond to all the already included contingencies.

3.4 Plan execution

If everything goes "according to plan", the agent simply executes a well-defined plan and, sometimes, contingencies arise and an agent reacts easily. Nevertheless, events that were not anticipated can happen at execution time, and for which there is no easy reaction to be applied. Thus, there are several different activities that are going on at execution time beyond simply executing a plan

1. **Reacting** to *unusual* events that can be predicted to happen from time to time.
2. **(Re)-planning** when the existing plan *will not cover the current situation.* Sometimes more than a simple course correction is required.
3. **Deciding** whether to re-plan or to react. This is similar to the decision problem that exists at planning time[12], but with a couple of additional complexities thrown in. When deciding at execution time, the additional decision of whether to abandon the existing plan must be made, and the decision process itself may be subject to real-time constraints that did not exist at planning time.

Anytime algorithms and real-time architectures are generally used to deal with these execution-time issues.

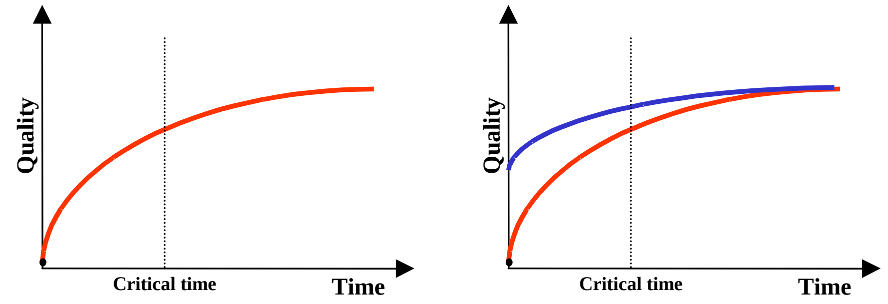
3.4.1 Anytime algorithms

The idea behind an anytime algorithm is that it produces an answer, no matter how long it runs. Whereas a conventional algorithm runs for a period of time, and then produces an answer, an anytime algorithm can be run for any amount of time and produces an answer. This answer will tend to improve over time. The idea here is that an anytime algorithm is used in cases where the amount of computational resources available to the algorithm is unknown at the time the algorithm begins running.

In anytime planning, the elaboration of a solution generally starts with a basic plan where the resources are allocated using a less sophisticated techniques, or simply randomly. Thereafter, it is improved by appropriately changing locally the resource allocation. Figure 4 (a) illustrates the augmentation of the quality of the plan according to time. One can see that, normally, the plan will gradually become better. On the other hand, at a certain given time, the agent can no longer improve the plan and must therefore act. This moment is represented on the graph by the critical time line. At this moment, the executed plan is not necessarily the optimal plan. But, in this case, a good plan now is better than a perfect plan later, since the cost may become too high if it waits too long. As shown in Figure 4 (b), there are two possible ways to improve the plan: 1) through the time; and/or 2) at the starting point. When time is critical, the solution should start with the best possible plan.

---

[12] To take in charge at planning time or to leave to execution time. And if taken at planning time, a decision must be made to use reaction or conditional branches.



(a) Quality of the plan according to time (b) Starting plan quality improvement

Figure 4: *Plan quality vs. starting plan and time*

3.4.2 Time-dependent planning

In the time-dependent planning framework [14], an agent has a number of different anytime algorithms it could work on at any given time. Each anytime algorithm corresponds to an event that occurs in the world demanding the agent's reaction. This would appear to involve an element of *meta-deliberation* in the sense that the agent has to spend some time making this decision, although the simplifying assumption that meta-deliberation time would be negligible was made. The agent's time is divided into *prediction time*, *deliberation time*, and *reaction time*. Prediction time refers to the time spent in predicting an event, given the information available. Presumably once the agent knows how to predict an event, it already knows the event is going to take place, and it can act accordingly. Deliberation time refers to the time available to the agent to determine a response to the event once it had occurred in the situation. Reaction time refers to the time taken to react to the event. In this framework, the key time is the *prediction time* since it allows the *meta-level* in the deliberation process to decide which anytime algorithm will be used to solve a particular problem.

4 Resource allocation planners

To achieve the present project objectives, three planning approaches for resource allocation were implemented and tested in the developed simulator (see Annex B). These include: Partly, Holistic Re-engagement, and Holistic Tabu planners. The three approaches use the cue generation (GENCUE) algorithm [15, 16] that constructs a first engagement list. Simple rules (purely reactive planner) are used for the deployment of softkill weapons. Hardkill and Softkill deployment decision processes, discussed in this section, are implemented by two separate planning agents. Interaction between agents is not discussed here, but documented in detail in [2].

The GENCUE algorithm is first presented then the internal representation of a defence plan will be shown. The planning algorithms used in the three hardkill defence modes will be introduced, then. Finally, all these planners will compared and the softkill algorithm will be described.

4.1 Cue generation algorithm (GENCUE)

This section presents how to produce a list of first engagements for the hardkill weapons using the cue generation (GENCUE) algorithm. GENCUE has been introduced by [15, 16], where an engagement is characterized by a defence resource, an illuminator (in the case of the SAM or the Gun), a target threat, and four time-stamped actions or cues, namely: (1) Search and lock on target; (2) Fire; (3) Target interception; (4) KA (*i.e.*, destroyed or not).

GENCUE (see Algorithm 1) constructs a list of feasible hardkill engagements, starting with the weapon that has the earliest target interception up time to the latest. It starts with an initial matrix E_0 of feasible time intervals and stops when all time intervals in the matrix are empty.

In this recursive algorithm, the procedure *Insert* generates and inserts cues associated with the engagement of threat j' by weapon i' for an interception at time $\tau(I_{i'j'})$. The procedure *Reduce* then produces a new matrix made of reduced time intervals that still obey the engagement doctrine; given that weapon i' now engages threat j'.

Furthermore, the selection of the latest feasible time to start an engagement is usually good practice, as the probability of interception for a defence weapon typically increases when the distance between the threat and the ship decreases (although this probability is assumed to be constant in this work, for the sake of simplicity). As the algorithm unfolds, a tree is generated, because many different engagements are possible at a given point through the choice of a particular weapon to fire or a particular threat to shoot at. A new conditional branch is thus generated for each possible engagement; the backward search is then applied in a recursive manner along each branch, until no feasible engagements can be found. The backward search is designed to maximize the number of re-engagements of a threat.

The following example shows how the cue generation is made with an example of two threats attacking the ownship. The data in Table 1 is obtained easily since the range is

Algorithm 1 GENCUE: Generation of the feasible hardkill engagement list

```
$t \leftarrow$ number of different weapons
$m \leftarrow$ number of threats
$E \leftarrow (I_{ij})$ is $t \times m$ matrix of feasible time intervals to intercept threat $j$ by weapon $i$

Function GENCUE($E$) return an engagement list $I$
$\tau(I_{ij}) \leftarrow \max(I_{ij})$ (i.e., the latest point in time over $I_{ij}$)
$\tau(E) \leftarrow \max_{i,j} \tau(I_{ij}) (= \max_{i,j} \max(I_{ij}))$
$f_{ij}(\tau(I_{ij})) \leftarrow$ firing time of weapon $i$ to intercept threat $j$ at time $\tau(I_{ij})$
if $E \neq 0$ then
  Select $(i^*, j^*)$ such that $(I_{i^*j^*}) = \tau(E)$;
  for each $(i', j') \big| \tau(I_{i'j'}) > f_{i^*j^*}(\tau(E))$ do
    $Insert(\tau(I_{i'j'}), i', j')$
    $E' \leftarrow Reduce(E, i', j')$
    GENCUE($E'$)
  end for
end if
return $I$
```

given directly from the radar. Then the ship hit time can be computed given that the threats travel at the assumed speed of 850 m/sec.

	Threat 1	**Threat 2**
Range (km)	28.31	8.70
Ship hit time (sec)	33.31	10.23

Table 1: *Threat specifications*

However, for Table 2, more advanced calculations are required.

	min/maxTime (Threat 1)	**min/maxTime (Threat 2)**
SAM	25.29/30.72	—
Gun	27.43/32.25	6.62/9.17
CIWS	30.36/33.31	7.29/10.23

Table 2: *Interception times for each type of weapon*

As an example, the following equations show how the intervals 27.43 ($minTime$) and 32.25

($maxTime$) for the Gun in Table 2 are obtained.

$$\begin{aligned}
minTime &= ShipHitTime - \frac{rangeMax}{GunSpeed} \times \frac{GunSpeed}{ASMSpeed} \qquad (1)\\
&= 33.31 - \Big[(5/0.850) \times (0.850/0.850)\Big]\\
&= 27.43\\
maxTime &= ShipHitTime - \frac{rangeMin}{GunSpeed} \times \frac{GunSpeed}{ASMSpeed} \qquad (2)\\
&= 33.31 - \Big[(.9/0.850) \times (0.850/0.850)\Big]\\
&= 32.25
\end{aligned}$$

Table 3 gives the variable values. Note that the information on threat specifications and interception times constitutes the input of the GENCUE algorithm. As soon as this information is available, GENCUE generates the list of feasible engagements.

Variable	**Value**	**Remarks**
$ASMSpeed$	850 m/s	
$GunSpeed$	850 m/s	
$rangeMin$	0.9 km	Min range to intercept a threat by Gun
$rangeMax$	5 km	Max range to intercept a threat by Gun
$minTime$		Min time of interception between the 2 weapons
$maxTime$		Max time of interception between the 2 weapons
$ShipHitTime$		Time the threat hits ownship

Table 3: *Variable values*

Engagement #	**Fire time [s]**	**KA time [s]**	**Threat**	**Weapon**
1	0.91	10.17	2	Gun-3
2	2.41	10.17	2	Gun-2
3	3.91	10.17	2	Gun-1
4	4.21	10.23	2	CIWS
5	19.49	33.25	1	Gun-6
6	20.53	33.22	1	SAM
7	20.99	33.25	1	Gun-5
8	22.49	33.25	1	Gun-4
9	23.99	33.25	1	Gun-3
10	25.49	33.25	1	Gun-2
11	26.99	33.25	1	Gun-1
12	27.28	33.31	1	CIWS

Table 4: *Engagements produced by the cue generation algorithm (GENCUE)*

To illustrate how such engagements are obtained, Threat 1 is considered, starting with the CIWS.

Variable	**Value**	**Remarks**
$rangeMax_{CIWS}$	$2.5km$	Max range to intercept a threat by CIWS
$CIWSSpeed$	$1200m/sec$	
$CIWSLockTime$	$1sec$	time for CIWS radar to lock on threat
$CIWSMinTime$		Min time of intercept obtained in Table 2
$CIWSAirTime$	$\frac{rangeMax_{CIWS}}{CIWS_{Speed}}$	Time CIWS will be in the air
$CIWSFireTime$		Time to start to fire CIWS
$GunFireTime$		Time to start to fire Gun
$CIWSSpace$	0.3 sec	Time that Gun shoots before CIWS

Table 5: *Variable values for Close-In Weapon System (CIWS)*

The variable values of Table 5 give

$$\begin{aligned} CIWSAirTime &= \frac{rangeMax_{CIWS}}{CIWS_{Speed}} \qquad (3) \\ &= \frac{2.5\ km}{1.2\ km/s} \\ &= 2.08\ s \\ CIWSFireTime &= CIWSMinTime - CIWSAirTime - CIWSLockTime \qquad (4) \\ &= 30.36 - 2.08 - 1\ s \\ &= 27.28\ s \end{aligned}$$

The fire time for the Gun can be obtained from the CIWS because the Gun cannot shoot the threat if the CIWS is engaging it. Consequently, here is how is computed the fire time of the first salvos of the Gun (Gun-1) for Threat 1.

$$GunFireTime = CIWSMinTime - CIWSAirTime - CIWSLockTime - CIWSSpace \qquad (5)$$

$$\begin{aligned} &= 30.36 - 2.08 - 1 - 0.3\ s \\ &= 26.98\ s \end{aligned}$$

Once this number is determined, 1.5 s is added between each salvo until it is impossible to intercept the threat at that time or the Gun would be out of range. For the SAM, the firing time upon the latest interception time is simply calculated and it is preceded similarly as for the CIWS.

To find the KA time for the weapons, the ship hit time is taken and the constant to evaluate the KA is withdrawn. Here is an example for the SAM (see engagement 6 in Table 4).

$$KASAM = ShipHitTime - KillAssSAM = 33.31 - 0.09 = 33.22\ s \qquad (6)$$

4.1.1 Plan representation

The generated plans are represented as directed decision tree, whose nodes are associated with fire and KA cues. A fire node has only one child in the tree, while a KA may have a number of children nodes, depending on the granularity of the threat destruction assessment. Furthermore, a "ship destroyed" node may be introduced in the tree at the time of impact of a threat with ownship if there is no action taken to counter the threat or if all actions taken to counter the threat fail. In the considered case, binary trees are constructed, as the threat kill and ship destroyed assessment are reduced to yes/no assertions (*i.e.*, kill/not kill or destroyed/not destroyed, respectively).

Figure 5 shows a small contingency plan constructed with some of the cues presented in Table 4. In this example, different geometric shapes are used to represent fire, kill and ship destroyed assessment nodes. The arcs are labeled with the possible outcomes of the kill and ship destroyed assessments.

Each path from the root to a leaf in such a tree represents a possible evolution of the situation in the real world. Thus, the leaves correspond to possible states of the world at the end of the planning horizon. A particular plan can be evaluated through a weighted sum over those leaves: each component of the sum is the Ownship Survival Probability (OSP) in the associated state, multiplied by the probability of ending that state.

The ownship survival probability at an internal node v, also known as quality or utility node v, can be recursively computed from its two children. Given u_1 and u_2 as the two children, the survival probabilities at the children nodes, u_v is then:

$$u_v = p \times u_1 + (1 - p) \times u_2 \tag{7}$$

4.2 Partly planner

First of all, a first engagement list is constructed with the help of the cue generation algorithm (see Section 4.1). Then, a plan called "Partly Plan" can be elaborated from the cues.

The Partly Planner uses very low-level reasoning techniques in order to elaborate a response to a situation in a very short reaction time. This is very important in combat RM context because defending ownship brings a very hard and usually very short time constraint. For this planning mode, the hardkill agent maintains a list of threats moving toward the ownship. This list is sorted (from the most to the least dangerous threat) according to some form of threat evaluation. For this implementation, threat evaluation considers only the Closest Point of Approach (CPA) of the threat to the ownship, and the time for the threat to reach CPA. Then, the hardkill agent applies some predefined rules for allocating the resources. These predefined rules are:

1. Allocate a SAM and the Gun to the most dangerous threat.
2. Allocate a SAM to the second most dangerous threat.

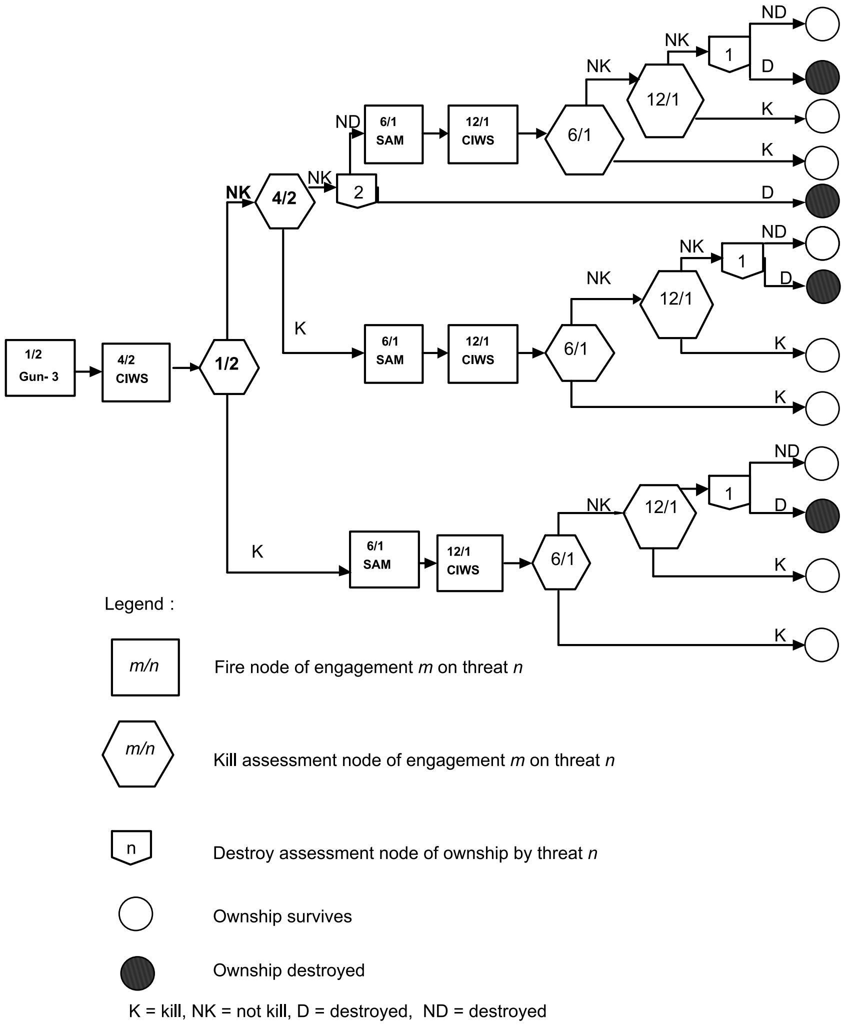


Figure 5: *Selecting and combining engagement cues as a tree (*i.e., *a contingency plan)*

3. Allocate the CIWS to all threats (one at a time) that enter the CIWS range.

The first two rules are inspired by the fact that there are only two STIRs available, which must be used in conjunction with the SAM and the Gun. The softkill planning is detailed in Section 4.5.

The “Partly Planner” is described by Algorithm 2. Though the rules used by the planner are simple, they allow all available resources to be used in an efficient way. Unfortunately, the SAM and the Gun are only allocated at a given point in time to the two most dangerous threats, and all other threats in the list (if any) are not considered by this specific planner (this is why it is called “Partly”). In the case where a KA indicates that a hostile threat has been destroyed, the resources that have been allocated to this threat become available for the next most dangerous threat in the list.

Algorithm 2 Partly planning for the hardkill weapons.

```
Function PARTLY-PLAN(ranked threats-list, $\mathcal{T}$) return a plan $\mathcal{P}$
$N \leftarrow$ number of threats in $\mathcal{T}$
$\mathcal{P} \leftarrow \emptyset$
for each i=1:2 do
  if it is possible to fire a SAM at the latest time possible to $\mathcal{T}(i)$ then
    $\mathcal{P} \leftarrow \mathcal{P} \cup$ SAM which is fired at the latest time against $\mathcal{T}(i)$
  end if
end for
if it is possible to fire GUN at the latest time possible against $\mathcal{T}(1)$ then
  $\mathcal{P} \leftarrow \mathcal{P} \cup$ GUN which is fired at the latest time against $\mathcal{T}(1)$
end if
for each i=1:N do
  if possible to fire CIWS against $\mathcal{T}(i)|i \neq 1 \& i \neq 2$ then
    $\mathcal{P} \leftarrow \mathcal{P} \cup$ CIWS which is fired against $\mathcal{T}(i)$
  end if
end for
```

Figure 6 provides the comparison for the "Partly Planner" when the SAMs are fired at the latest and when they are fired at the earliest time possible.

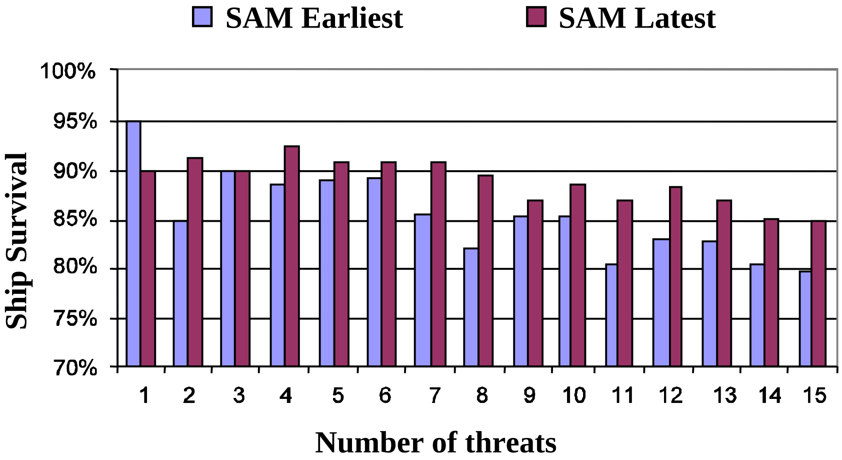


Figure 6: *Partly planning by firing SAMs at the earliest or latest time possible*

One can notice that firing SAMs at the latest time possible is more efficient (by an average of 4.4%) than firing SAMs at the earliest time. Firing in the earliest mode is normally used just after the radar of the ownship perceives the threat. The difference of effectiveness between the two methods is due to a better RM, because the STIR is less used when the missile is fired at the latest time possible. There are only two STIRs to control SAMs, so they must be used very carefully, and they must not guide a SAM during a too long period. During this time, they cannot do anything else. Therefore, firing SAMs at the latest time possible helps to optimize combat resource utilization.

This planner can be improved in different ways. The next section presents an improved version where, the planner takes into consideration all threats instead of the two first ones. Also, this version allows for the re-engagement of same threat by the SAM in the case of a miss.

4.3 Holistic re-engagement planner

A first list of engagements is obtained with the cue generation algorithm as discussed in Section 4.1. As this planner should consider all threats and it can re-engage with a SAM, it is called the hardkill Holistic Re-engagement Planner. In the cue generation, there is a key difference in the SAM engagements from the cues generated for the Partly Planner. Instead of inserting one SAM engagement at the latest possible time, it can use re-engagements of the SAM against a threat. Indeed, if after the KA of the first SAM the targeted threat is still alive, another SAM can be engaged to destroy it. Usually, there will be at least one re-engagement for each threat. In the planning process, the engagements of the SAM are scheduled backward in time from the latest time of fire possible. SAMs are added in the Holistic Re-engagement Planner until it is not possible anymore to do the KA of the current SAM nor to be able to engage the next one. Another difference from the Partly Planner is that the Holistic Re-engagement Planner considers all visible threats to make a plan.

This planner views all the detected threats constituting a complex setting surrounding the ownship. It works as follows: a decision tree is first produced that explicitly considers, in a probabilistic manner, all possible outcomes of a particular action. Such a tree reflects in fact a plan with different conditional branches. That allows to take into account results of actions. For instance, during the plan execution, one should follow one branch or another depending on the result of an engagement to some threat $\mathcal{T}(i)$. If this engagement has succeeded, then the plan continues by following a branch where it does not consider the threat $\mathcal{T}(i)$ anymore. If the engagement has failed, then a branch where other engagements are planned for $\mathcal{T}(i)$ is executed. All these conditional branches reflect contingent plans that are very important since the outcomes of the engagements are uncertain (see Subsection 3.3.3). Notice that without conditional branches, the time horizon of the plan would be very limited, and it would be needed to re-plan each time an engagement fails. The latter can take a long time, thus causing problems for the subsequent engagements.

The Holistic Re-engagement Planner is given by Algorithm 3. It uses the following rules:

1. the closest threats are engaged first. Note that the threats are ranked based on their distance from ownship, where the closest threat has the rank 1;
2. a SAM has priority over the Gun to engage a threat;
3. the CIWS engages whenever possible;
4. the number of re-engagements of a threat is maximized.

The root is a dummy node that initially contains the list of all engagements identified by the cue generation algorithm. As the algorithm unfolds, compatible engagements are inserted in the plan, using the engagement compatibility subset of the valid node with the earliest time stamp. When its subset becomes empty, the procedure is repeated with the next valid node. For the sake of simplicity, the above procedure assumes that each node has a single engagement subset. However, the extension for nodes with two subsets is quite straightforward: both subsets must be emptied before proceeding with the next valid node.

Algorithm 3 Holistic Reengagement planner for the hardkill weapons

```
Function HOLISTIC-PLAN( threats-list, $\mathcal{T}$) return a plan $\mathcal{P}$
Create a root node (cue algorithm with SAM re-engagement)
$C$ is the subset of engagements identified by the cue generation algorithm
$C_v$is the subset of engagements compatible with node $v$
$t_v$ is the time stamp of node $v$ (i.e., time of the associated cue)
$C_{root} \leftarrow C$
$t_{root} \leftarrow$ start of the planning time horizon
while there is a valid node $v$ in the plan (i.e., exist $v$ that $C_v \neq 0$) do
  select the valid node $v$ with earliest $t_v$
  $r \leftarrow 1$
  scan $C_v$ until one of the following is found (in this order)
      SAM engages the threat with rank $r$
      Gun engages the threat with rank $r$
      CIWS engages the threat with rank $r$
  if an engagement is found then
    insert it in the plan $\mathcal{P}$ and update the engagement compatibility subsets
  else
    $r \leftarrow r + 1$
  end if
  if $r \leq m$ (number of threats) then
    go back to "scan $C_v$ until one of the following is found (in this order)"
  end if
end while
return $\mathcal{P}$
```

4.4 Holistic Tabu Planner

Like the Holistic Re-engagement Planner, the Holistic Tabu Planner works with a decision tree to execute the plan, and it considers all visible threats for the plan. However it does not use the SAM re-engagements, but instead uses Tabu heuristics for improving the quality. Therefore, it uses Algorithm 3 to create the initial tree. After that, it uses Algorithm 4 to improve it. More precisely, the initial tree is improved by a Tabu search [15] through the removal or addition of defensive actions, followed by update operations aimed at maintaining the consistency of the plan. Note that in recent years, Tabu search has been applied with a high degree of success to a variety of problems. It is based on an iterative neighborhood search method where modifications to the current solution that degrade the solution value are admissible. The latter move allows the method to escape from local optima (as opposed to a pure local search approach). To avoid cycling, a short-term memory, known as the Tabu list, stores previously visited solutions or components of previously visited solutions. It is then forbidden or “Tabu” to come back to these solutions for a certain number of iterations.

Algorithm 4 Holistic Tabu planner for the hardkill weapons

```
Function TABU-PLAN(threats-list, $\mathcal{T}$) return a plan $\mathcal{P}^*$
$\mathcal{P} \leftarrow$ HOLISTIC-PLAN($\mathcal{T}$)
$\mathcal{P}^* \leftarrow \mathcal{P}$
while stopping criteria of tabu search is not met do
  $\mathcal{P}' \leftarrow NonTabuNeighborhood(\mathcal{P})$
  if $Consistent(\mathcal{P}')$&$\mathcal{P}'$ is better than $\mathcal{P}^*$ then
    $\mathcal{P}^* \leftarrow \mathcal{P}'$
  end if
  $\mathcal{P} \leftarrow \mathcal{P}'$
  Update Tabu list
end while
return $\mathcal{P}^*$
```

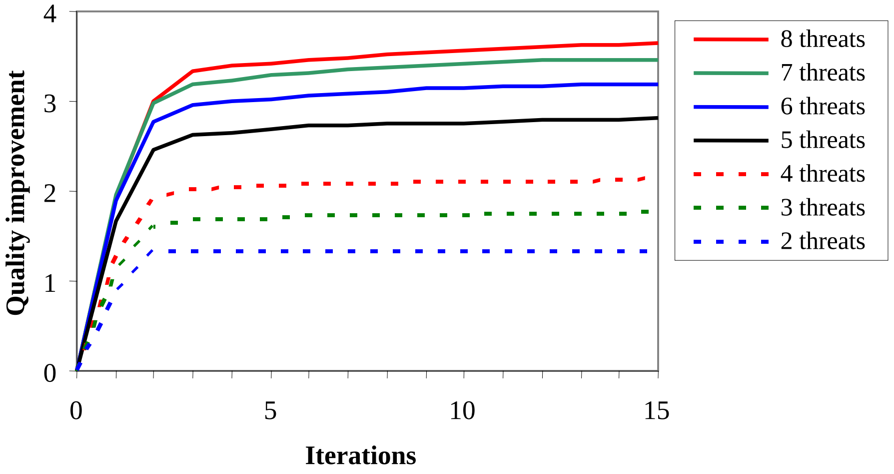


Figure 7: *Tabu Planner - Plan quality improvement*

Figure 7 shows how the quality of response increases with planning time, for different numbers of threats to face. Figure 8 compares, for Tabu and Holistic planners, the planning time as a function of the number of threats (and the number of iterations for the Tabu).

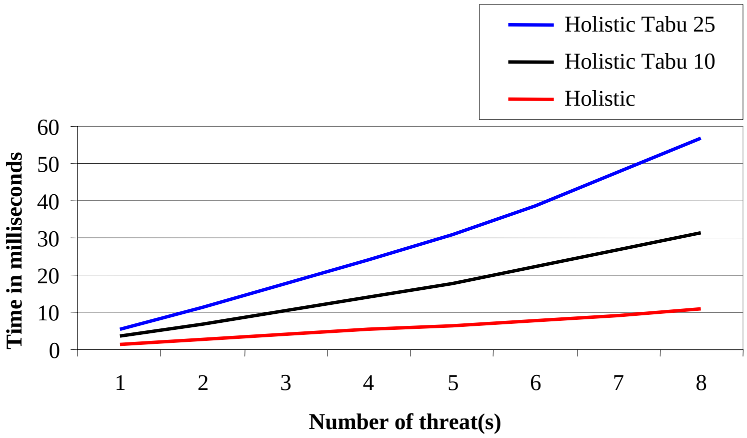


Figure 8: *Response time of planners (Tabu N = Tabu with N iterations)*

4.5 Softkill Planner

The used Softkill Planner is reactive and is implemented by the softkill agent. This agent manages two types of resources: jamming and chaff. For the application of interest, there are two jamming units and four Chaff launchers. Jamming units can act on two threats each. Starting from these considerations, the softkill agent elaborates a Partly or Holistic softkill plan. To do that, it starts from the list of threats attacking the ship (sorted by order of importance, from the most to the least dangerous). Then the planner applies a simple rule that consists of allocating a jamming unit and a chaff launcher in order to address all possible threats (Holistic) or the new detected threat(s) (Partly). Figure 9 shows the result of combining the reactive "Softkill Planner" with the different hardkill strategies.

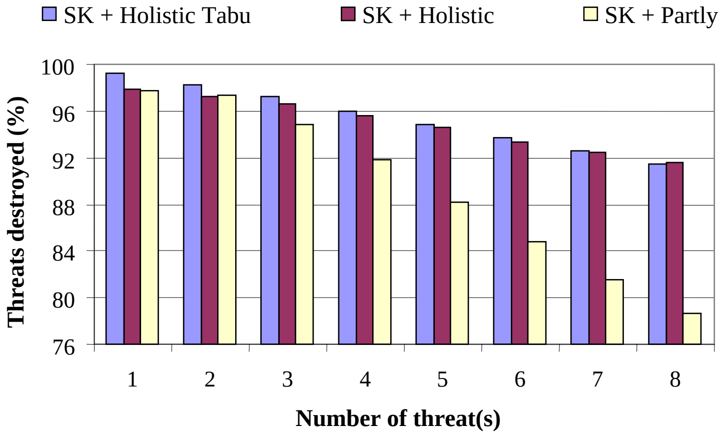


Figure 9: *Softkill planner performance in combination with the different Hardkill planners*

4.6 Comparison of planners

Figures 10 and 11 compare the Partly and Holistic Re-engagement planners when only softkill weapons are used, only hardkill weapons, or when the two weapons systems are coordinated to face diverse attack scenarios. Two remarks need to be made here.

1. In the case of Softkill Agent, the Holistic Planner is a reactive (rule-based) one that is different from Holistic Re-engagement used by the Hardkill Agent.
2. The issue of hardkill/softkill coordination is beyond the scope of this report and is treated in [2].

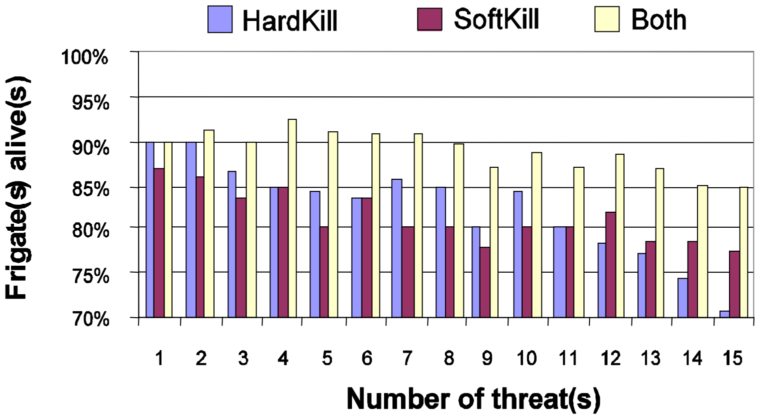


Figure 10: *Partly planning system using hardkill, softkill, or both weapons*

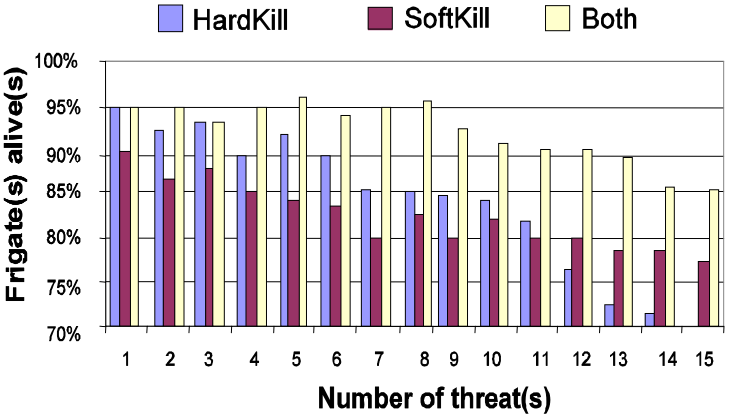


Figure 11: *Holistic Re-engagement planning using hardkill, softkill, or both weapons*

With regard to the distinction between reactive planning (Partly Planner) and deliberative planning (Holistic Planner) in the case of hardkill, the preliminary results show that the deliberative plans are generally more effective than the reactive plans (as indicated in Figure 12). These results also show that the effectiveness of the deliberative plans degrades more quickly than the reactive plans when the number of threats increases. These results can be explained by the fact that, at some point, the time of deliberation becomes too high, and consequently, the agents do not have enough time to build good deliberative plans.

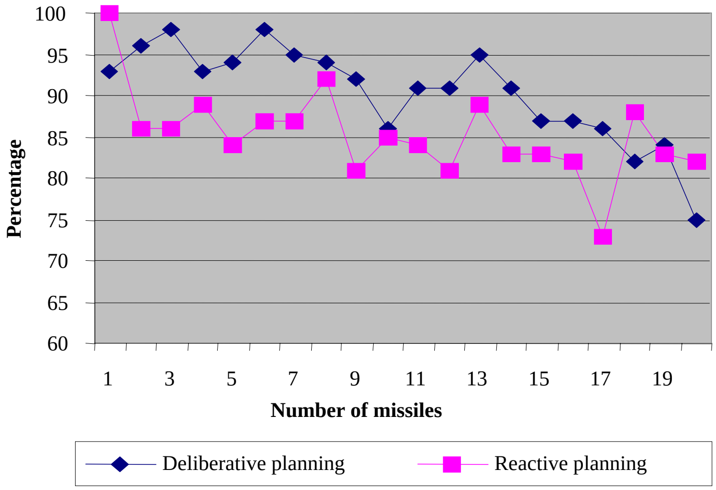


Figure 12: Reactive (Partly) versus deliberative (Holistic) hardkill planning

Some remarks can be made on the basis of these results.

1. The global efficiency of the softkill and hardkill weapons is almost similar. The hardkill mode is better with few threats but less effective taking account of lots of threats than the softkill mode. In face of few threats, the ownship has more chances of survival because: (1) it can use the CIWS against a higher percentage of threats. When the CIWS is used against two or three threats, there are no more units available to face another threat, which is less effective with a scenario with a lot of threats; (2) as there are only two STIRs available to control SAMs, this causes difficulties when facing more than two threats simultaneously. When the softkill weapons are used alone, the Chaff is used more than when the two types of weapons are coordinated.

   In this situation, it is possible to assign a chaff launcher and a jamming chanel to each threat, resulting in a higher survival chance for the ownship. The used model assumes the availability of 30 units of Chaff and the absence of restriction on the capacity to use them. The jamming has more usability constraints than the Chaff, but less than the STIRs. There are two jamming antennas that can each face two threats. Hence it provides more channels (four) to allocate to threats than the two STIRs available for the hardkill weapons. Therefore, it can be seen that the softkill weapons are more flexible to face more threats than the hardkill weapons. For the used scenarios in general, similar results were obtained because when the hardkill weapons can be used without resource constraints, they are more effective than the softkill weapons. This occurs with scenarios having few threats. On the other hand, softkill weapons are more effective to face more threats.

2. Using both the softkill and hardkill weapons improves the survival chances for the

ownship. One can obviously see that there is a synergy effect in the coordinated use of the two types of weapons. In general, the original hardkill plan is used along with the jamming in the global planning, which is better than the use of the hardkill or the softkill only (see [2] for more details).

3. Softkill results of the Holistic Re-engagement mode are slightly better than the Partly mode. Although this difference is small, it should be considered because the two softkill algorithms are similar aside from the fact that it is planned for all visible threats for the Holistic Re-engagement mode.

4. Hardkill results for the Holistic Re-engagement mode are slightly better than the Partly mode. The improvement is due to the use of SAM re-engagements for the Holistic Re-engagement mode. Although this difference is generally weak, it is not the same in all specific situations, as explained in the following remark.

5 Conclusion

The reported work is part of a project that aimed at contributing to the effort of acquiring knowledge and expertise in the use of multi-agent technology to address tactical resource allocation planning and coordination problems. The outcome of this effort can be integrated into the ship's Command and Control System (CCS) to assist operators in conducting the tactical C^2 process, focusing on Above Water Warfare (AWW). Planning with multi-agent systems has been studied in this project with the main objective of conceiving a combat resource allocation and coordination capabilities for a generic warship. This could ultimately be extended to the Halifax and Iroquois Classes, or even the future Single Class Surface Combatant (SCSC), platforms. Allocation problem algorithms were presented and discussed in this report. The coordination problems are discussed in the companion report [2].

The developed Agent and Multi-agent-based allocation and coordination algorithms were implemented in the test-bed environment. The developed simulator considers a generic ship and uses a simplified model of the relevant Ant-Air Warfare (AAW) hardkill and softkill weapons. This model was a simple, unclassified version of AAW hardkill and softkill for the Halifax Class Frigate, but did preserve the fundamental features of these weapons to evaluate different planning and coordination algorithms and approaches.

In this report, the theoretical basis and definitions of Agent and Multi-Agent Systems (MAS) were introduced. Then, planning problems were presented in a multi-agent context. Both planning-time and execution-time approaches, to handle contingencies, were discussed. These approaches include reaction, conditional planning, and replanning. A real-time issue was considered through anytime algorithms. The engagements are first generated with a cue generation algorithm (GENCUE). Then, the three defence planners, which are "Partly", "Holistic Re-engagement", and "Holistic Tabu", were presented. These planners were tested and their respective advantages and disadvantages discussed.

Deliberative planners, that is "Holistic Re-engagement" and "Holistic Tabu", showed a slight superiority over the purely reactive one (*i.e.*, "Partly"). Note that the latter is very close to the way the planning is performed within the current versions of the Canadian Navy CCSs. Even tough these are only preliminary results, which required further investigation and validation, they show the potential improvement that can be brought to the CCSs by using more deliberative planning techniques.

5.1 Recommendations

During the course of this project, the following areas of investigation arose. They were beyond the current scope of work, but should be considered for future projects.

1. The global planning framework used in this project can obviously be greatly upgraded. For example, when a plan is about to be implemented, if a new threat is detected, a new global plan for every visible threat can be made using dynamic replanning. As

an alternative, it may be faster and more efficient to repair the original plan using either reactive rules or one (or more) of the conditional branches in the original plan.

2. There are a variety of interesting problems associated with deliberative planning. For example:
   (a) assessing which parameters should be considered, and how they should be weighted, in the utility function used for evaluating and selecting sequences of plan actions;
   (b) investigating the choice of optimization algorithm for searching through the "space" of possible plan actions, in order to determine the "optimal" plan (only Tabu search has been considered so far); and
   (c) exploring whether (or not) there is a need (and how) to generate multiple plans in parallel for different hypotheses of real-world state, and how to evolve these into an active plan.

3. The Partly and Holistic Re-engagement planning algorithms may not need to be anytime compliant because their goal is to provide a single and fast solution. Although, the Holistic Tabu planner continues improving its own plan for a period of time, a plan is available at any time. Based on the observations from experiments, more work needs to be done to improve and thoroughly investigate the Holistic Tabu planner. Thus, all the planners could be subject to modification and upgrade.

4. It may be useful to investigate having combat RM agents controlled or directed, at least in part, by a hierarchical structure of influences that are for the most part *a priori* knowledge. This would encompass, for example, rules of engagement, standard operating procedure, doctrine, and tactics.

5. It was noticed that the Holistic Re-engagement planner is clearly better than the Partly planner when few threats attack the ownship, but both planners offer similar results when many threats attack. So, it could be interesting to have a meta-level agent that decides which kind of planner to use according to the situation. This could be implemented using a meta-deliberation technique.

6. Only AAW hardkill and softkill weapons were considered in this project. The solutions presented have endeavored to accommodate, within their general architecture, future expansion to include other weapon systems (*e.g.*, ASuW[13] and ASW[14]), but no explicit implementation of these weapon systems was performed.

7. Of necessity, the weapon specifications and threat scenarios used in this project were quite simple. To lend greater credibility and usefulness to the results, it would be beneficial to apply more complex weapon and threat models and more complicated threat scenarios. For example, the threat scenarios used for this project all had 0.0 Closest Point of Approach (CPA) with respect to the frigate, which effectively minimizes the value of the Holistic planners. Realistic scenarios would not have all threats with $CPA = 0.0$.

---

[13] Anti-Surface Warfare

[14] Anti-Submarine Warfare

This project contributed to four primary research efforts. The first one is to develop a multi-agent approach for tactical RM. The second is to allow university and industry partners to acquire knowledge and expertise in the use of (i) multi-agent planning, (ii) multi-agent scheduling, (iii) decision support systems and (iv) agent technology for RM. The third, which is a long-term effort, is to allow DRDC to apply the knowledge gained, and to utilize the algorithms developed in a variety of applications. Finally, the ultimate effort is to contribute to the development of methodological knowledge and skills so much needed in Canada to meet the challenge of decision making in the context of RM.

Even though it brought several contributions on the algorithmic side, the report project focused mainly on the architectural aspects and simulator development. It used simplified prototypes to demonstrate the usefulness of the planning and coordination mechanisms for the applications to naval RM. However, before this technology can be fully utilized in these applications, there are still fundamental research questions to be answered and problems to be investigated. This is why a follow-on project has been initiated, the main objective of which is to bring the investigated concepts and technology closer to real-world and very complex Command and Control System (CCS) application.

This page intentionally left blank.

References

[1] Duquet, J. R., Bergeron, P., Blodgett, D., Couture, J., Chalmers, B. A., and Paradis, S. (1998), Analysis of the Functional and Real-time Requirements of a Multi-agent Data Fusion (MSDF-STA-RM) Systems, In *SPIE Proceedings*, Vol. 3376, pp. 198–202.

[2] Benaskeur, A., Bossé, É, and Blodgett, D. (2006), Multi-Agent Coordination techniques for Naval Tactical Combat Resources Management. DRDC Valcartier TR 2006-784. UNCLASSIFIED.

[3] Ahuja, Ravindra, Kumar, Arvind, Jha, Krishna, and Orlin, James (2004), Exact and Heuristic Methods for the Weapon Target Assignment Problem, (4464-03). available at http://ideas.repec.org/p/mit/sloanp/7388.html.

[4] Hosein, P. A. and Athans, M. (1990), Some Analytical Results for the Dynamic Weapon-Target Allocation Problem, *Naval Research Logistics*.

[5] Jennings, N., Sycara, K., and Wooldridge, M. (1998), A Roadmap of Agent Research and Development, *Autonomous Agents and Multi-Agent Systems*, 1(1), 7 – 38.

[6] Wooldridge, M. and Jennings, N. (1995), Intelligent agents: Theory and practice, *The Knowledge Engineering Review*, 10(2), 115–152.

[7] Russell, S. J., Subramanian, D., and Parr, R. (1995), Provably Bounded Optimal Agents, In *International Joint Conference on Artificial Intelligence 1995*, Vol. 2, pp. 575–609, Montréal, Canada.

[8] Boosh, G. (1994), Object-Oriented Analysis and Design (second edition), Reading, MA: Addison-Wesley.

[9] Agha, G. (1986), ACTORS: A Model of Concurrent Computation in Distributed Systems, MIT Press.

[10] Agha, G., Wegner, P., and Yonezawa, A., (Eds.) (1993), Research Directions in Concurrent Object-Oriented Programming, MIT Press.

[11] Maes, P. (1994), Agents that Reduce Work and Information Overload, *Communications of ACM*, 37(7), 31–40.

[12] Bond, A. H. and Gasser, L., (Eds.) (1988), Readings in Distributed Artificial Intelligence, Morgan Kaufmann.

[13] Durfee, E. H. and Lesser, V. (1989), Negotiating Task Decomposition and Allocation Using Partial Global Planning, In Gasser, L. and Huhns, M., (Eds.), *Distributed Artificial Intelligence (DAI)*, Vol. 2, pp. 229–244, Morgan Kaufmann.

[14] Dean, T. L. and Boddy, M. (1988), An analysis of time-dependant planning, *Proceedings of the 7th National Conference on Artificial Intelligence (AAAI-88)*, pp. 49–54.

[15] Blodgett, D., Gendreau, M., Guertin, F., Potvin, J.-Y., and Séguin, R. (1998), A Tabu Search Heuristic for Resource Management in Naval Warfare, (Technical Report #CRT-98-63) Centre for Research on Transportation, Université de Montréal.

[16] Chalmers, B. A. and da Ponte, P. (1995), MIMD Algorithms for Naval Resource Planning: Overview and Preliminary Assessment, Technical Report DREV-R-9420 Defence Research Establishment Valcartier, Québec, Qc, Canada.

[17] Rhéaume, F. and Benaskeur, A. (2004), Impact of the Tactical Picture Quality on the Fire Control Radar Search-Lock-on Time. DRDC Valcartier TR 2003-284. UNCLASSIFIED.

[18] Blodgett, D., Plamondon, P., Chaïb-draa, B., Kropf, P., and Bossé, E. (2002), A Method to Optimize Ship Maneuvers for the Coordination of Hardkill and Softkill Weapons within a Frigate, In *Proceedings of the 7th International Command and Control Research and Technology Symposium (7th ICCRTS)*, Québec, QC, Canada.

[19] Plamondon, P., Chaïb-draa, B., Beaumont, P., and Blodgett, D. (2003), A Frigate Movement Survival Agent-Based Approach, In *Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'2003)*. University of Oxford, United Kingdom.

Annex A: Tactical shipboard combat resources

The present Halifax Class Frigate (see Figure A.1) design provides for a layered response to threats. The Surface-to-Air Missile (SAM) system operates first for targets at medium ranges, while the other two systems take over at closer ranges, first the Medium Caliber Gun (MCG) then the Close-In Weapon System (CIWS). The Halifax Class Frigate weapon system also includes softkill capabilities (decoy and jamming). Note that only Anti-Air Warfare (AAW) resources are discussed.

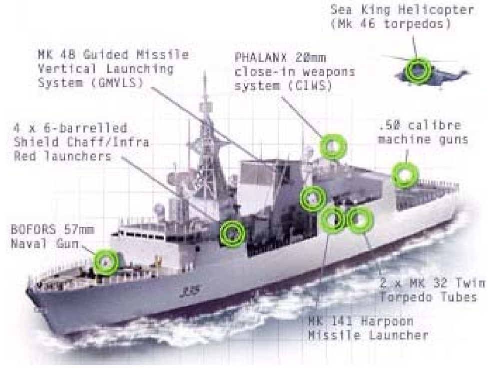


Figure A.1: *Halifax Class Frigate*

The Frigate's hardkill weapons (SAMs, Gun, CIWS) are used in conjunction with specialized radars used to aim and/or guide weapons to targets. The SAMs and intermediate range gun are supported by a Separate Tracking and Illuminating Radar (STIR) and the CIWS by the CIWS radar. These weapon systems are described in the following sections.

A.1 Assumptions

The exact nature of the specifications and capabilities of the various AAW hardkill and softkill weapons on the Halifax Class Frigate is obviously very complex, and much of that information is classified. In order to avoid the procedural complications of using classified information, and to maintain emphasis on the allocation and coordination techniques and not be burdened by the complexity and fidelity of the representation of hardkill and softkill, a considerably simplified model of the relevant AAW hardkill and softkill weapons was used. This model is a simple, unclassified version of AAW hardkill and softkill for the Halifax Class Frigate, but does preserve the fundamental features of these weapons. This "generic" frigate thereby becomes the basis for the investigations conducted in this project. The details of the model for hardkill and softkill are provided below.

A.2 Surface-to-air missile (SAM)

Presently, the primary weapon against air threats is the Vertically Launched Sea Sparrow (VLSS). The Halifax Class Frigate's Sea Sparrow GMVLS is capable of intercepting medium-range airborne targets (and horizon range surface targets), thanks to eight vertically mounted twin-canister launchers for surface-to-air engagement of hostile targets. There are eight Mk 48 launchers port and starboard.

Position on the ship : The SAM is, for simplicity, considered to be positioned at the centre of the ship $(x, y, z) = (0, 0, 0)$. In subsequent work, a position that reflects its actual emplacement on the ship will be considered.

Blind zone: It is assumed that the SAM has no blind zone for launching, but the blind zone of the associated STIR will affect the engagement space.

Kill Probability: When the target falls within the missile's effective range, the probability of kill for a SAM is assumed constant (set at 65%). In the first phase, and for simplicity, this will be assumed independent of target position. Later, the target position could be taken into consideration, and the kill probability will be redefined as a function of the target position with respect to the ship position.

Range: The range of the SAM is about 10 nautical miles. For simulation purposes, it will be assumed a maximum range $R_m^+ = 16$ km and a minimum range $R_m^- = 1.5$ km. These ranges are assumed constant for any bearing or elevation.

Speed : The SAM is assumed to travel following a ballistic (straight line) trajectory at the constant speed Mach 1 ($\equiv$ 340) m/s.

Unit Cost: A cost C_m per missile is assumed.

Launcher : There are eight Mk 48 launchers port and starboard. It is assumed that there are 16 SAMs, all initially loaded, and no replacements are available. There will be no delay between the time when the fire order is issued and that when the missile is launched. This assumption may be reconsidered in a later phase.

Guidance System: The Sea Sparrow is a semi-actively guided missile that homes in on targets illuminated by the Continuous Wave Illuminator (CWI) radar that is associated with the STIR.

Fire Control : At least one STIR fire control radar must be operational and the threatening targets must not fall within STIR's blind zone for Sea Sparrow interception to be feasible. A SAM can be fired only after a STIR has locked on the target. Provided STIR acquisition has been achieved, the SAM system addresses only targets at medium ranges.

A.2.1 Constraints and consequences

Given the characteristics of the weapons and the parameters of the scenario, different constraints and information will be required.

1. **Maximum Interception Range:** A target is detected and being tracked by the search radars beyond the maximum range of the STIR. The STIR is cued by a search

radar immediately at its maximum range R^+_{STIR}, and it begins its search without delay. Assume that the STIR will take t_{sl} seconds to acquire and lock-on the target. If a SAM is fired as soon as lock is obtained, the following equation gives the maximum range R^+ at which the interception will occur.

$$\begin{aligned} R^+ &= \left[R^+_{STIR} - V_t \times t_{sl}\right] \times \frac{V_m}{V_m + V_t} \\ &= (50.0 - 0.85 \times 3.0) \times \frac{0.34 \times 0.9}{0.85 + 0.34 \times 0.9} \\ &= 12.6\ km \end{aligned} \tag{A.1}$$

Where:

- R^+: the maximum range to intercept target
- R^+_{STIR}: the maximum range of the STIR
- V_t: the target speed
- t_{sl}: the STIR search and lock-on time
- V_m: speed of SAM

2. **Maximum Target Speed:** A special case of the previous calculation is when $R^+ = R^-_m$. This corresponds to the maximum of the speed of the target that the SAM can intercept. This is entirely defined by the properties of the STIR.

$$V^+_t = \frac{(R^+_{STIR} - R^-_m) \times V_m}{R^-_m + V_m \times t_{sl}} \tag{A.2}$$

A.3 Medium caliber gun

The Bofors SAK 57 L/70 Mk 2 GWS is an unmanned, all-purpose, rapid-fire Medium Caliber Gun (MCG) that can engage both aircraft and anti-ship missiles at close range. It is a 57 mm caliber gun that is very effective against surface targets out to horizon range, although it is usually used closer in.

Position on the ship: For simplicity, the gun is considered to be positioned at the centre of the ship $(x, y, z) = (0, 0, 0)$. In future versions, a position that reflects its actual emplacement on the ship will be considered.

Range: a maximum range of 5.0 km and a minimum range of 0.9 km are assumed.

Rate of Fire: The gun is capable to fire up to 200 rounds/min. These can be fired either in a single shot or in burst mode (n_s). The gun can fire consecutive salvos. The only constraint is that a KA must be performed for each salvo, and will take place after the last round in a salvo reaches the point of interception with the target. Also, in the case of firing several consecutive salvos, allow for the possibility of reassigning the associated STIR (and issuing a cease fire order if it is not too late) based on a kill observation from one of the intermediate KAs before the last round in the last salvo has reached the point of interception with the target. The gun fires in salvos that can be set to 1 to 10 rounds (assume for now that consecutive salvos can be fired with no delay between them). After every 30 shots (*i.e.*, two 15 round magazines), it takes 5 s to reload the magazines, but the magazines can be reloaded anytime there are 7 rounds or fewer remaining (note that none of the remaining rounds are lost in the reloading process – expended rounds are just replaced). The total number of rounds available in one load of the gun is 150; the gun can be completely reloaded (*i.e.*, providing another 150 rounds) in 8 minutes, with a total of 750 rounds available at the start of the mission. However, these capabilities are further simplified in the implementation of the different planning algorithms according to:

1. The gun fires only with a salvo length of 5 rounds.
2. The gun can fire consecutive salvos.
3. The salvo will not be fired, if there is not time to fire all rounds in a salvo.
4. Schedule of a KA for each salvo is established, which takes place after the last round in a salvo reaches the point of interception with the target.
5. A contingency plan for reassigning the associated STIR (and issuing a cease fire order if it is not too late) is allowed in the case of firing several consecutive salvos. This is based on a kill observation from one of the intermediate KAs before the last round in the last salvo has reached the point of interception with the target.

   Note — In this context, it is possible to fire several consecutive salvos without waiting to confirm KA from a prior salvo before firing the next. This is a slight relaxation of shoot-look-shoot doctrine.
6. The reloading time for the gun is 0 s, and can take place in mid-engagement.

   Note — Performing a complete reload of the gun is ignored, since given the specifications of threat generation, scenarios will last much less than the required 8 minute reload time. Consequently, there is an effective limit of 150 gun rounds available.

Muzzle Velocity: The gun rounds travel following a ballistic (straight line) trajectory at the constant speed of $V_g = 850$ m/s.

Training rate: For simplicity, the slew time to move the gun into position to fire at the target is assumed null. In future more sophisticated version of the scenario a training rate of $50°/s$ will be considered.

Magazine Capacity: The total number of rounds available for the gun is 500 rounds (one minute of continuous firing).

Unit Cost: A cost C_g per round is assumed.

Launcher: There are four ammunition racks in the turret, each with 32 rounds. It is assumed that there is no delay between the time the fire order is issued, and the gun starts shooting.

Blind Zone: Since there is only one intermediate range gun placed in the front of the ship, it has a blind zone of ±35° in azimuth looking in the backward direction. To this blind zone, one must add the blind zones imposed by the allotted STIR.

Kill Probability: It will be assumed that the probability of kill for the gun is $P_{K_R} = 0.04/\text{round}$. As for simplicity, it is assumed constant and independent of target position, as long as the latter is within the gun effective range. In a future version the the probability, of kill will be redefined as a three-phase (increasing, constant, decreasing) function of the distance between the target and the ownship. The probability of kill when the maximum number possible of rounds $N_{R_{max}}$ is fired at a threat is given by:

$$P_{K_{NR}} = 1 - \left[1 - P_{K_R}\right]^{N_{R_{max}}} \tag{A.3}$$

Fire Control: The gun and SAM share the same fire control radars. The STIRs must be operational for the MCG to be guided and the gun can be fired only after a STIR has locked on the target.

A.3.1 Constraints and consequences

As in the case of the SAM, the characteristics of the MCG weapons and the parameters of the scenario impose different constraints.

1. **Maximum Intercept Range:** A target is detected and being tracked by the search radars beyond the maximum range of the STIR. The STIR is cued by a search radar immediately at its maximum range R^+_{STIR}, and it begins its search without delay. Assume that the STIR will take t_{sl} seconds to acquire & lock on the target. If the gun is fired as soon as lock is obtained, the following equation gives the maximum range R^+ at which the interception will occur.

$$\begin{aligned} R^+ &= \left[R^+_{STIR} - V_t \times t_{sl}\right] \times \frac{V_g}{V_g + V_t} \qquad \text{(A.4)} \\ &= (50 - 0.85 \times 3.0) \times \frac{0.85}{0.85 + 0.85} \\ &= 23.7km \end{aligned}$$

   Where:
   - R^+: the maximum range to intercept target
   - R^+_{STIR}: the maximum range of the STIR
   - V_t: the target speed
   - t_{sl}: the STIR search and lock-on time
   - V_g: speed of the MCG

2. **Maximum Duration of Firing:**

$$\begin{aligned} D_{F^+} &= \frac{R_{GUN^+}}{GUN_s} + \frac{R_{GUN^+} - R_{GUN^-}}{T_s} - \frac{R_{GUN^-}}{GUN_s} \qquad \text{(A.5)} \\ &= \frac{5.0}{0.85} + \frac{5.0 - 0.9}{0.85} - \frac{0.9}{0.85} \\ &= 9.65\ s \end{aligned}$$

where D_{F^+} is the maximum duration of firing, R_{GUN^+} is the maximum range of the gun, and R_{GUN^-} its minimum range. Note that:

(a) R_{GUN^+}/GUN_s is the lead time.

(b) $(R_{GUN^+} - R_{GUN^-})/T_s$ is the flight time of the threat while in the range of gun.

(c) R_{GUN^-}/GUN_s is the time for weapon to reach minimum weapon range.

The time to fire a complete magazine of 30 rounds is

$$30/(200/60) = 9.0\ s$$

With a maximum of only 9.65 s. for firing, after firing 30 rounds the time remaining (0.65 s) is less than the 5 s. required to reload the gun magazines. Consequently, a maximum of 30 rounds can be fired at a target by the gun.

Finally, the probability of kill per round (P_{K_R}) is chosen such that

$$P_{K_{Nmax}} = 1 - \left[1 - P_{K_R}\right]^{Nmax} \tag{A.6}$$

where $Nmax$ is the maximum number of rounds and $P_{K_{Nmax}}$ is the probability of kill for maximum number of rounds.

A.4 Close-in weapons system (CIWS)

The Phalanx Mk 15 Mod 1, a Close-In Weapon System (CIWS), provides the Canadian Navy ships with a terminal point defence capability. It is a self-contained, search, detect, track, and engage weapon system that can be targeted, based upon the Command and Control System (CCS) input or operated in a fully automatic mode. The CIWS provides an ultra-high fire rate of 20 mm shells that represents a "last chance" protection against anti-ship missiles, fixed-wing aircraft (and surface targets) that may have penetrated the ship's outer defence systems (at very close range).

Currently, the Phalanx's primary role is mainly considered to be the detection and the automatic engagement of low-level, pop-up anti-ship missile attackers. Target kill/survival assessments are an important feedback that can be provided by the CIWS to the whole CCS. Also can be provided to the CCS are track data, considering the fact that the CIWS has its own detection 3-Dimension tracking system and can be fused as another sensor.

Position on the ship: For simplicity, the CIWS is considered to be positioned at the center of the ship $(x, y, z) = (0, 0, 0)$. In future versions, a position that reflects the actual emplacement of the CIWS on the ship will be considered.

Range: It is assumed a maximum range of 2.5 km and minimum range of 0.0 km.

Speed: The CIWS rounds travel following a ballistic (straight line) trajectory at the constant speed of $V_c = 1200$ m/sec.

Rate of Fire: The gun is capable to fire up to 55 rounds/s.

Magazine Capacity: The total number of rounds available for the CIWS is 1500 rounds (one minute of continuous firing).

Fire Control: The CIWS can be fired only after the CIWS self-contained search and track radar. has locked on the target. It will be assumed that the slew time to move the CIWS into position to fire at the target is 0.0 s and there is no delay between the time the fire order is issued and the CIWS starts shooting.

Blind Zone: Due to its emplacement at rear of the ship, the CIWS suffers from a blind zone of ±15° in bearing looking in the forward direction, and from 70° – 90° in polar angle (where 90° is vertical).

Kill Probability: It will be assumed that the probability of kill for the CIWS is $P_{K_R} = 0.006$/round. As for the simplicity, this is assumed constant and independent of target position, as long as the latter is within the CIWS effective range. In a future version the probability of kill will be redefined as a three-phase (increasing, constant, and decreasing) function of the distance between the target and the ownship. The probability of kill when the maximum number possible of rounds $N_{R_{max}}$ is fired at a threat is given by:

$$P_{K_{NR}} = 1 - \left[- P_{K_R} \right]^{N_{R_{max}}} \tag{A.7}$$

A.4.1 Constraints and consequences

1. **Maximum Interception Range:** It is assumed that the CIWS fire control radar starts its search immediately at its maximum range R^+_{CIWS} and that it will take t_{sl} seconds to acquire and lock on the target. If the CIWS is fired as soon as lock is obtained, the following equation gives the maximum range R^+ at which the interception will occur.

$$\begin{aligned} R^+ &= \left[R^+_{CIWS} - V_t \times t_{sl} \right] \times \frac{V_c}{V_c + V_t} \\ &= (5.5 - 0.85 \times 1.0) \times \frac{1.2}{0.85 + 1.2} \\ &= 2.7 \ km \end{aligned} \tag{A.8}$$

   Where

   - R^+ : the maximum range to intercept target
   - R^+_{CIWS} : the maximum range of the CIWS fire control radar
   - V_t: the target speed
   - t_{sl}: the CIWS fire control radar search and lock-on time
   - V_c: speed of the CIWS rounds

2. **Maximum Duration of Firing:**

$$
\begin{aligned}
D_{F^+} &= \frac{R_{CIWS^+}}{CIWS_s} + \frac{R_{CIWS^+} - R_{CIWS^-}}{T_s} - \frac{R_{CIWS^-}}{CIWS_s} \\
&= \frac{2.5}{1.2} + \frac{2.5 - 0.0}{0.85} - \frac{0.0}{1.2} \\
&= 5.0\ s
\end{aligned}
\tag{A.9}
$$

where D_{F^+} is the maximum duration of firing, R_{CIWS^+} is the maximum range of the CIWS and R_{CIWS^-} is its minimum range. Thus, the maximum number of rounds possible is $5.0 \times 55.0 = 275.0$.

A.5 Separate tracking and illuminating radar (STIR)

The current Halifax Class Frigate combat system comprises two Fire Control Radar (FCR) systems called Separate Tracking and Illuminating Radar (STIR: respectively, STIR-A and STIR-B). The STIRs are responsible for the control of the Surface-to-Air Missile (SAM) fire channels and the Medium Caliber Gun (MCG). They provide the SAM and MCG weapon systems with fire control quality track data for engageability and fire control calculations.

The system also provides designation to the Close-In-Weapon-System (CIWS) for targets when the CIWS is designated to engage an air target. Because there are two STIRs (and hence two fire control channels) available, the combat RM system can launch either one or a salvo of two missiles against the highest threat target, and then almost immediately the same against the second highest inbound threat. The STIRs provide the capability of tracking air (and surface) targets ranges such that all ownship weapons can be launched to intercept their targets at maximum weapon range.

Inputs to STIRs are lock-on target information (position) and commands. Outputs are status reports. Care must be taken to prevent both STIRs from engaging a single threat. Appropriate delays for STIR acquisition before firing must be part of the planning mechanism of combat RM [17]. The main characteristics of the STIRs are as follows.

Position on ownship: For simplicity, both STIRs (STIR-A and STIR-B) are considered positioned at the center of the ship $(x, y, z) = (0, 0, 0)$. In future versions, positions that reflect the actual emplacement of the STIRs on the ship will be considered.

Blind zone of STIR-A: ±60° in azimuth, in the backward direction.

Blind zone of STIR-B: ±60° in azimuth, in the forward direction.

Range: Each STIR is assumed to have an effective range of 50 km. Both units are assumed to have polar angle ϕ coverage of $0° < \phi < 90°$

Search and lock time: For simplicity, it is assumed to be constant at $t_{sl} = 3.0$ s. In future versions, the search and lock time will depend upon the quality of the track provided to the STIR by the search radars [17].

Hand-off: Once STIR lock is obtained, control can be passed to the second STIR with presumably no delay (*i.e.*, the second STIR is provided with a precise bearing and elevation in order to begin its search). Note that control is passed to a second STIR only if there is no SAM in-flight (*i.e.*, once a SAM is launched, it must be guided to the target by the same STIR assigned to it at the launching time). STIR must remain illuminating the target during total time-of-flight of SAM to target. If the STIR controls a gun, the STIR can be unlocked at any time, causing the gun to cease firing.

KA: It is assumed that the KA performed via the STIR for SAM takes a fixed duration of 2.5 s. But it is assumed the KA done via the STIR for the gun takes a fixed duration of 1.0 s.

A.6 Close-in weapons system (CIWS) radar

Unlike the STIR, there is only one unit that is entirely dedicated to CIWS. The time for the CIWS radar to search for and lock on the target is assumed constant and set at 1.0 s. In the autonomous mode of CIWS operation, the CIWS is assumed to independently search for targets, so that there is always a finite search and lock time for the CIWS radar to acquire a target. Note that there is another mode of operation where, if a STIR is locked on a target, the CIWS can be given that precise position to start its search phase. The time to do so is negligible, and it leads to directed firing on a specific target.

The CIWS illuminator can be unlocked at any time, causing the CIWS to cease firing. The KA performed via the CIWS illuminator takes a fixed duration of $1sec$ (but for now, KA for the CIWS will be ignored).

1. **Blind zone:** The same blind zone as the one of the CIWS described in Section A.4, that is $\pm 15^\circ$ in bearing looking in the forward direction. The main CIWS FCR characteristics are given below.
2. **Range:** It is assumed to have a maximum range of 5.5 km and minimum range of 0.0 km.
3. **Estimate of search and lock-on time:** In the autonomous mode of CIWS operation, the time for the CIWS radar to search for and lock on the target is assumed to be constant, set at 2.0 s. When the CIWS radar is cued by the STIR (that provides it with a precise position to begin its search and lock) the time is negligible, then set at 0.0 s.

A.7 Softkill combat resources

The Halifax Class Frigate combat resource also include Electronic Attack (EA) capabilities. EA has the mission to prevent or reduce the enemy's use of electromagnetic (EM) spectrum. It has, for many years, been called Electronic Counter-Measures (ECM). EA uses EM or directed energy to attack personnel, facilities, or equipment in order to:

1. damage physically the enemy assets by use of high levels of radiated power or directed energy; this is referred to as destructive EA; and
2. make the enemy asset temporarily ineffective but does not destroy it. This is rather a non-destructive EA, also referred to as "soft kill". This form of EA is the only one that will be considered in this project.

The softkill weapon suite on the Halifax Class Frigates comprises: a jammer, decoys (Chaff, Sonobuoy, Rubber duck) and flare.

A.7.1 Jammer

A jammer aims at modifying the waves of the radar that controls the Anti-Ship Missile (ASM) that comes toward the ownship. It tries to modify the destination of the ASM by affecting its own radar. There are two primary modes of use for jamming:

1. Break missile lock on ownship — Assume a 20 s duration for the jammer to search for and acquire the missile threat, and then process to cause the missile to break its radar lock on ownship.
2. Create a false target position on the missile's radar — A jam pulse is used to create a delayed offset from a normal radar reflection, what is interpreted by the missile's radar as the actual target position. Because of the offset, the range determined by the missile's radar is greater than the actual range of the target. Once the jammer has acquired the missile (*e.g.*, in the break lock mode), this processing happens quickly, say in 3.0 s.

The maximum jamming range is 25 km . The percentage of success for jamming only (without chaff) is 40% and the percentage of success for jamming and chaff together is 80%. There are two antennas. One is assumed to see ±100° pointing left of ownship, the other ±100° pointing right of ownship (*i.e.*, there are regions of overlap, of 20° each, between the two antennas. One is located at the front of the ship, the other at the back). Each antenna can deal with up to two threats. Responsibility for jamming a threat can be passed from one antenna to the other, provided, of course, that the threat is within the new antenna's coverage region when it is to take over, and that the new antenna is not already dealing with two other threats.

A.7.2 Decoys (Chaff)

A decoy is designed to look from the enemy seeker[15] perspective more like a protected platform than the protected platform itself. It aims at causing the guided weapon to attack the decoy rather than its intended target, for instance. The difference between decoys and jammers is that decoys do not interfere with the sensors tracking them, but rather seek to attract the attention of those sensors causing them either to, acquire it and attack it, or to transfer the tracking focus.

---

[15] Be it operating in Radio-Frequency (RF) or Infra-Red (IR) frequency bands.

The Chaff bursts constitute the main decoy system against radar-guided threats for the Halifax Class Frigate. Chaff bursts may be used as both an expendable distraction or seduction decoys for ship protection against ASMs. In this case, the separation of the decoy from the target is generated only by the movement of the ship and by the wind, which moves the chaff burst. The chaff burst is ideally placed in the corner from which it will separate from the ship most rapidly. The burst placement is chosen based on the type of radar in the attacking missile, the relative wind direction and velocity, and the direction from which the attack is coming.

For this project, a maximum Radar Cross Section[16] (RCS) of 5000 m^2 is assumed for the chaff. Also, for simplification, it is assumed that the chaff cloud instantaneously forms a sphere that remains fixed in size for the duration of the cloud. At a later time, one can add time for the cloud to plume, use a more realistic shape for the cloud, and account for degradation and movement of the cloud due to gravity and environmental effects.

- Deployment range (from ownship): 225 m
- Duration of chaff : up to 10 min
- Total inventory of chaff shells: 30
- % of success for chaff only (no jamming) : 30%
- % of success for jamming and chaff together: 80%

In the current chaff system (SHIELD), target information (position, velocity, etc.), meteorological conditions, and an instruction to deploy chaff are all input to the system, so the latter determines when and where the chaff will be deployed. Note that the chaff will be greatly affected by meteorological conditions, which have not been accounted for in this description.

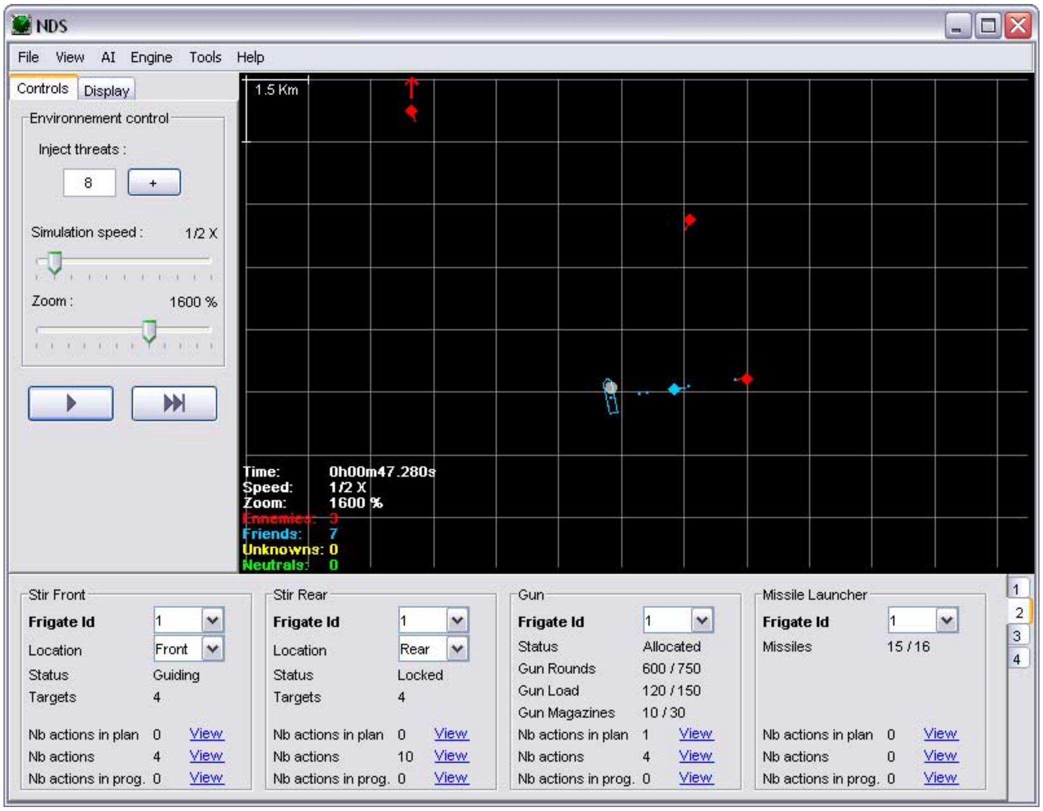
---

[16] The RCS is the measure of a target's ability to reflect radar signals in the direction of the radar receiver.

This page intentionally left blank.

Annex B: Naval defence simulator (NDS)

A simulator has been developed during this project to allow, through various scenarios, for large amount of tests on the investigated agent and multi-agent-based planning concepts. The Naval Defence Simulator (NDS) shown in Figure B.1 allows specific tests to be replicated as many times as desired, what is obviously impossible to match on real-life systems. With low costs compared to real-life demonstrations, this allows to develop, implement, validate, and compare a broad range of concepts. Another advantage of having a simulator is that it allows us to focus on particular aspects of the C^2 process.



Inject threats : 8
Simulation speed : 1/2 X
Zoom : 1600 %

Time: 0h00m47.280s
Speed: 1/2 X
Zoom: 1600 %
Ennemies: 3
Friends: 7
Unknowns: 0
Neutrals: 0

Stir Front
Frigate Id 1
Location Front
Status Guiding
Targets 4
Nb actions in plan 0
Nb actions 4
Nb actions in prog. 0

Stir Rear
Frigate Id 1
Location Rear
Status Locked
Targets 4
Nb actions in plan 0
Nb actions 10
Nb actions in prog. 0

Gun
Frigate Id 1
Status Allocated
Gun Rounds 600 / 750
Gun Load 120 / 150
Gun Magazines 10 / 30
Nb actions in plan 1
Nb actions 4
Nb actions in prog. 0

Missile Launcher
Frigate Id 1
Missiles 15 / 16
Nb actions in plan 0
Nb actions 0
Nb actions in prog. 0

Figure B.1: *Naval Defence Simulator*

In the project, the focus has been on resource allocation and coordination, so the situation analysis problem has not been solved.

B.1 Architecture

The programming language used is Java, because of its ease of use, flexibility, and portability. The NDS is developed in three-tier architecture, as shown in Figure B.2. The first tier, the *Data*, is composed of the simulation objects. The second tier, the *Logic*, is composed of many subsystems and is responsible for the kinetics, time flow, agents, and communications management. When an object needs to be inserted (*e.g.*, when firing a SAM) or deleted (*e.g.*, when an ASM has been destroyed), it is the task of the *engine* to evaluate the relevance of the action and take the appropriate steps. The last tier, the *User Interface*, is the medium of interaction between the users and the engine. It is with the Graphical User Interface (GUI) that users create and record scenarios, get a view of the internal values of objects and start batch tests. The design of the simulator itself makes it easy to deactivate the GUI and use automated test modes.

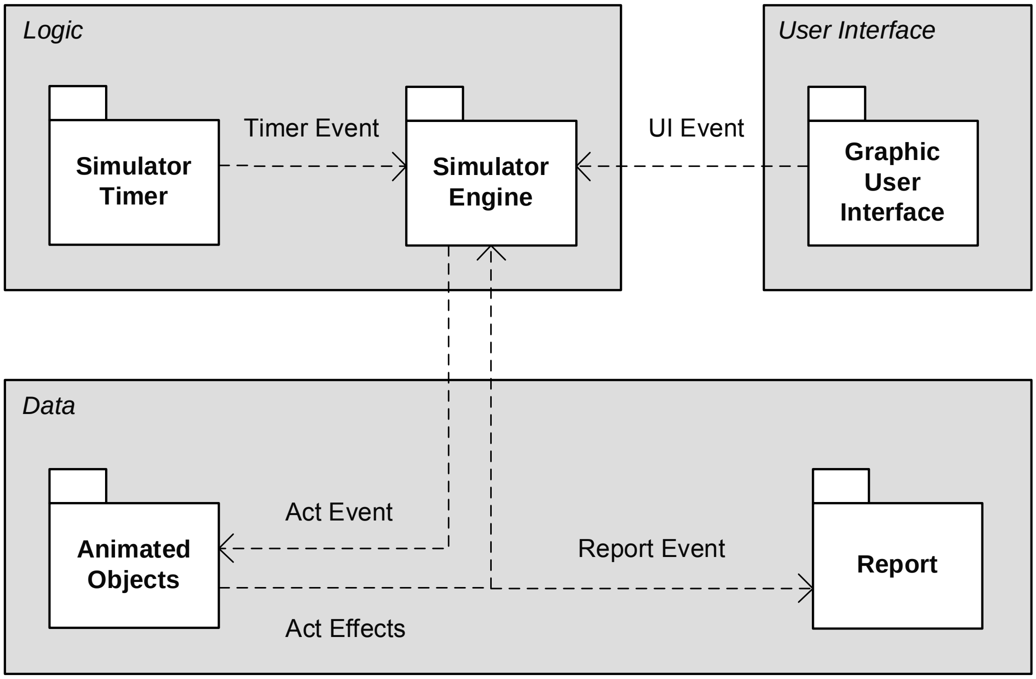


Figure B.2: *Naval Defence Simulator architecture*

B.1.1 Time management

Discrete time mechanism is used as a simulation approach. In this structure, every object then has, for acting, the same virtual time to act. The *timer* triggers time events in the engine, which then runs each object for a specific time quantum. Once an object is run by the engine, it acts, deletes impossible actions and moves. After all objects have moved, collisions are evaluated and destroyed objects are cleared from the simulation. While acting, objects look for valid actions in their list to execute. Software planners take the extra step of planning which actions should be executed beforehand. This is illustrated in Figure B.2.

An interesting advantage of this mechanism is that one can easily speed up or slow down the simulation. There are two factors that can be changed: the interval at which time events are sent, and the time quantum in which each object must act. Thus, it is virtually possible to speed up the simulation to hundreds of times faster than real-time, but still can let some part execute in real-time when necessary (as with some anytime algorithms). In fact, when sped up to its maximum value, a typical simulation of 5 minutes lasts less than 1 second on the computer used to test scenarios. Furthermore, the simulator has been designed in such a way that it is possible to vary the simulation speed while leaving the normal Central Processor Unit (CPU) time to the planning algorithms.

B.1.2 Agent management

In the current version of the simulator, there is one agent for each ship[17] that is responsible for deliberating on the situation of its attributed ship. In the NDS, each object, including agents, act for a specified time quantum during each simulation *round*. The inner control loop of the various agents let them monitor their environment and plan on the evolving situation. Moreover, they can receive messages and react at any time during a simulation to a more complex situation. These messages are received through the *Communication Central*.

B.1.3 Communication management

In Mutli-Agent Systems (MAS), cooperating agents often need to exchange messages. In the NDS, this is done via the *Communication Central*, which receives messages from agents and dispatches them to the correct recipients. Mostly, it serves to model communication waves in the simulator environment and accordingly delays the reception of messages by the receiving agents. Three different delays are introduced for each message sent.

- *Message preparation delay* – This is a constant delay, representing the time needed to ready the physical communication channel and prepare the message by wrapping it with the appropriate headers.
- *Distance induced delay* – This is the delay induced by the physical distance between sender and receiver. This is derived from the speed of radio waves that is $300,000$ m/s. Therefore, the beginning of a message sent to an ally 3 km away will be received 10 milliseconds later.
- *Bandwidth induced delay* – This is the delay induced by the total length of the message. By varying this parameter, it is possible to simulate various communication conditions, such as to simulate a stronger encryption, thus reducing the bandwidth and decreasing the total throughput of the system. Even though this is not yet implemented, an appropriate reduction in bandwidth when jamming units are used can be simulated by modeling the background noise of the system.

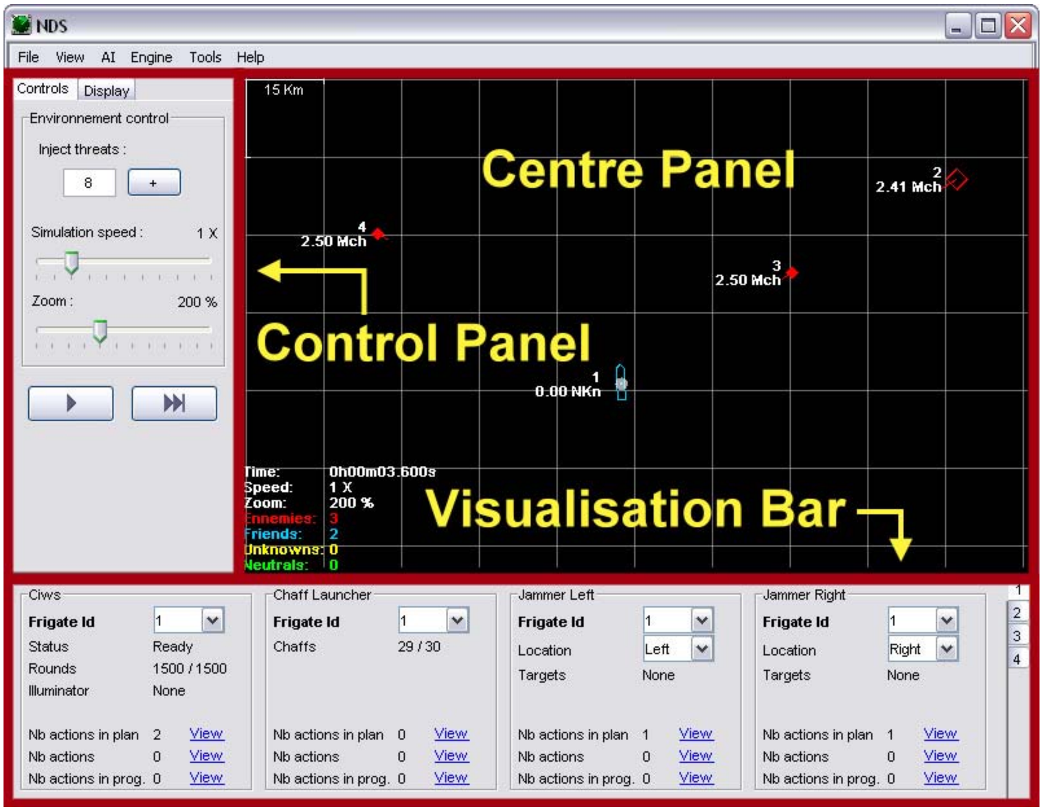
B.2 User interface panels

On the left side of the GUI is a zone with two different panels. The first one is the simulation control panel, which contains elements such as speed and zoom controls. The simulation progress can be entirely controlled from this panel, or also from the engine menu and keyboard shortcuts. In the simulation control panel, a user can:

---

[17] However, it is possible to have more than one agent for each ship. For example, it is possible to use one agent for hardkill systems and one for softkill systems. It is also possible to have specific agents responsible for multi-platform coordination, etc.

- add new threats generated at random position,
- start and pause the simulation at any given time,
- zoom in and out between 12.5% and 25,600%,
- speed up or slow down the simulation between 1/4X and 256X, and
- advance the simulation by exactly one turn, which is equal to 80 milliseconds in simulation time.



Environnement control

Inject threats : 8 +

Simulation speed : 1 X

Zoom : 200 %

Time: 0h00m03.600s
Speed: 1 X
Zoom: 200 %
Ennemies: 3
Friends: 2
Unknowns: 0
Neutrals: 0

Ciws
Frigate Id 1
Status Ready
Rounds 1500 / 1500
Illuminator None
Nb actions in plan 2 View
Nb actions 0 View
Nb actions in prog. 0 View

Chaff Launcher
Frigate Id 1
Chaffs 29 / 30
Nb actions in plan 0 View
Nb actions 0 View
Nb actions in prog. 0 View

Jammer Left
Frigate Id 1
Location Left
Targets None
Nb actions in plan 1 View
Nb actions 0 View
Nb actions in prog. 0 View

Jammer Right
Frigate Id 1
Location Right
Targets None
Nb actions in plan 1 View
Nb actions 0 View
Nb actions in prog. 0 View

Figure B.3: *Components of the Graphical User Interface (GUI)*

The centre panel of the simulator allows users to follow the development of the current simulation. It uses symbols and colors to visually represent objects. Table B.1 shows the different symbols used in the simulation. The color code (Table B.2) serves to represent the allegiance of the objects.

The following objects, for visibility concerns, derogate from this color code. First, the cargo vessels are represented in white, to further accentuate the fact that they are units with no actual defensive capability. Second, the chaff clouds are represented as white circles with alpha blending.

The left zone also contains a second panel that is the simulation display panel. With this panel, the user controls display on the centre panel. There are display options common

Symbol	Object
⬠	Cargo vessel
⬠	Frigate
◇—	Airplane
◆—	Missile
●	chaff cloud

Table B.1: *Symbols used in Naval Defence Simulator (NDS)*

Color	Allegiance
Blue	Allies
Red	Enemies
Yellow	Unknown
Green	Neutral

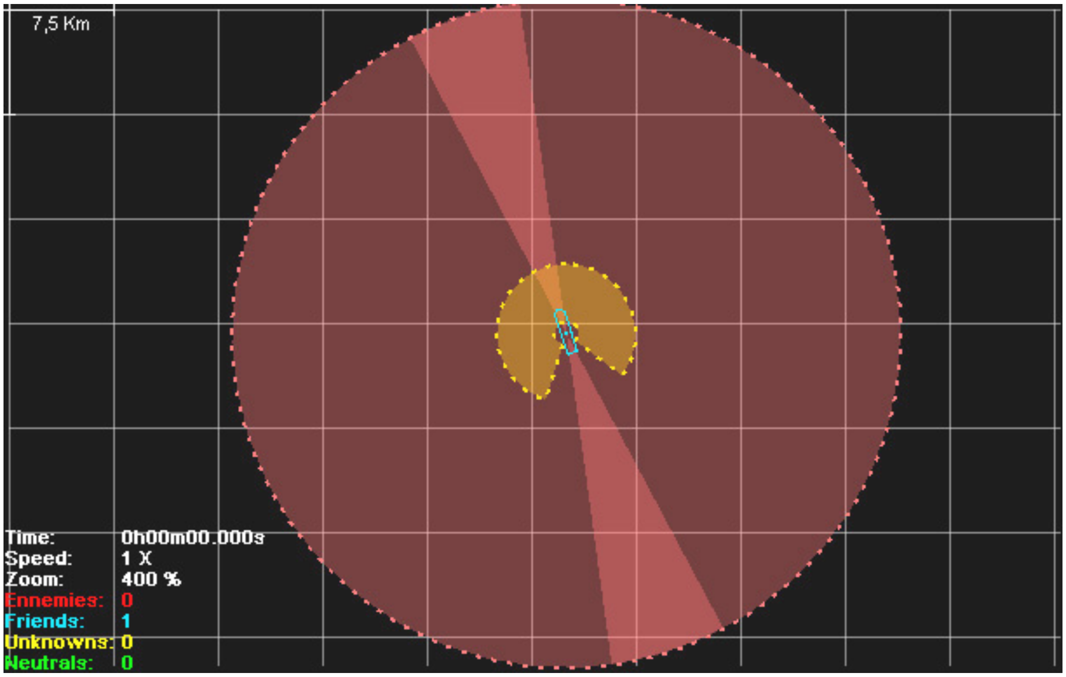
Table B.2: *Color codes used in Naval Defence Simulator (NDS)*

to most objects and some other options applicable only to specific objects. The elements of the first category allow displaying the unique Identity (ID) of an object, as well as its speed and position. In the central panel, gun and Close-In Weapon System (CIWS) rounds and chaff clouds will not have their information displayed. The reason for this is that there would be too much information packed in the same space and it would clutter the display with no appreciable added value. On the other side, the elements specific to ships allow the user to display simultaneously the coverage range of any onboard system, such as radar and CIWS. Figure B.4 shows the range/blind zone of the gun as well as the coverage zone of both jammer units on each side of the ship. Note that areas where both jammer systems are available are clearer.

The bottom panel of the GUI contains the visualization bars. There are no real limits on the number of bars that can be present. Each bar is customizable and contains four visualization items. A right click on any item slot lets the user change what is shown in this slot. The implemented visualization items are separated in three types, as depicted in Table B.3.

System	Objects	Frigate modules
Memory Consumption System Information	Airplane Missile Frigate Cargo vessel	CIWS Gun Missile Launcher Chaff Launcher Jammer STIR

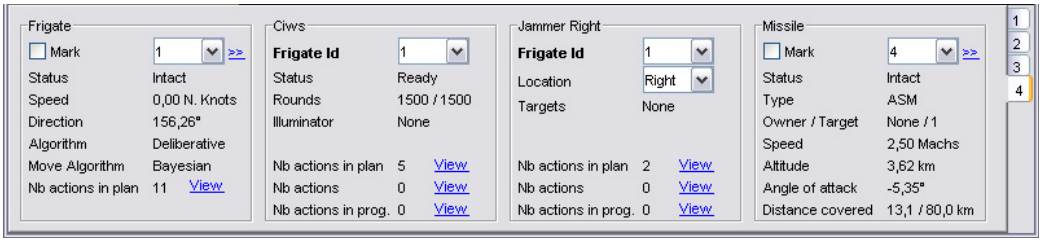
Table B.3: *Available visualization items in NDS*



Time: 0h00m00.000s
Speed: 1 X
Zoom: 400 %
Ennemies: 0
Friends: 1
Unknowns: 0
Neutrals: 0

Figure B.4: *Example of system ranges*

The first class contains the items pertaining to the simulator core. The second class of items is the object visualization items, which present information relative to specific objects and object types. The last visualization item is specific to NDS and shows the different resource modules of the frigates. It is easy for a developer to create new visualization types. Moreover, the settings chosen and the displayed visualization bars are saved when changed and reloaded when the simulator is launched again. Figure B.5 shows a visualization bar with four visualization items (in this case a *Frigate View Item*, *CIWS View Item*, *Jammer View Item* and a *Missile View Item* items).



Frigate

☐ Mark	1
Status	Intact
Speed	0,00 N. Knots
Direction	156,26°
Algorithm	Deliberative
Move Algorithm	Bayesian
Nb actions in plan	11

Ciws

Frigate Id	1
Status	Ready
Rounds	1500 / 1500
Illuminator	None
Nb actions in plan	5
Nb actions	0
Nb actions in prog.	0

Jammer Right

Frigate Id	1
Location	Right
Targets	None
Nb actions in plan	2
Nb actions	0
Nb actions in prog.	0

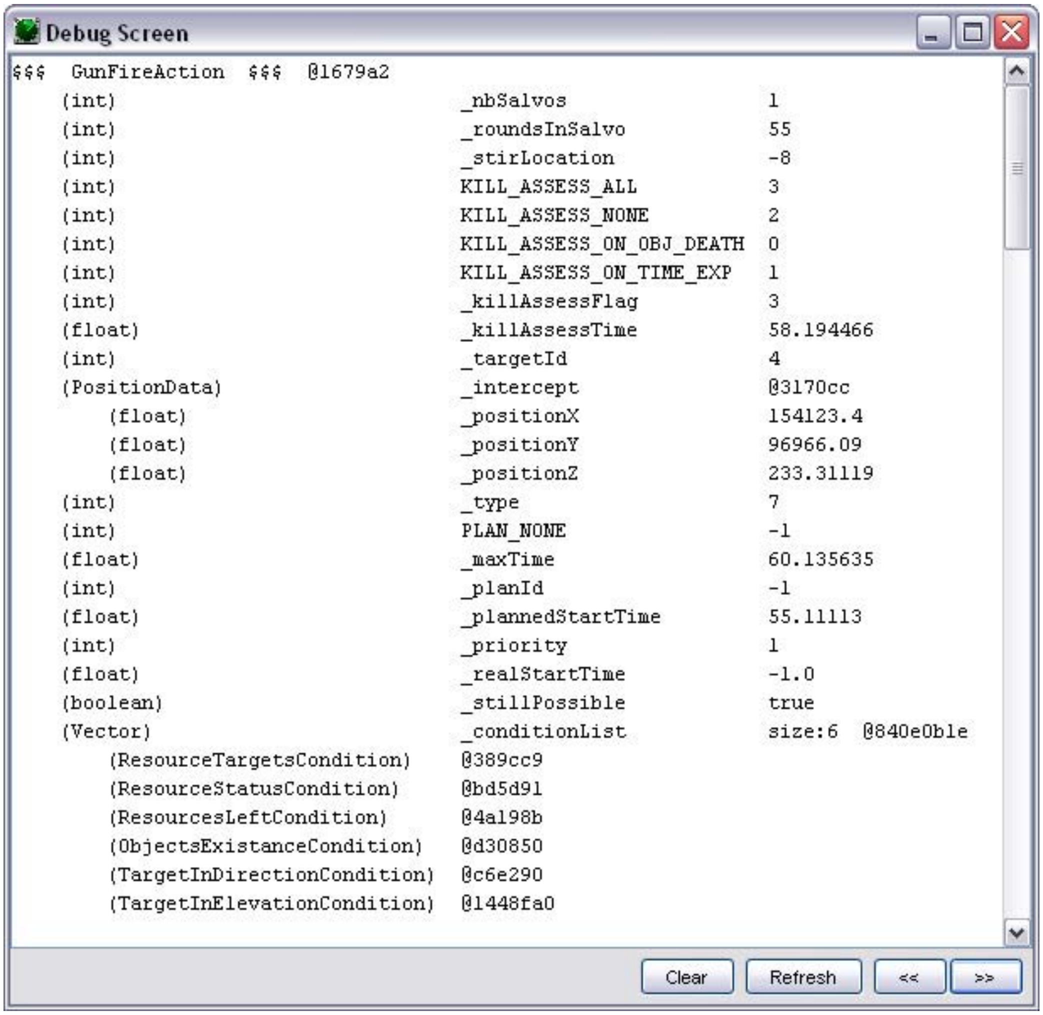
Missile

☐ Mark	4
Status	Intact
Type	ASM
Owner / Target	None / 1
Speed	2,50 Machs
Altitude	3,62 km
Angle of attack	-5,35°
Distance covered	13,1 / 80,0 km

Figure B.5: *Bar of visualization items*

B.3 Debugging

Included for developers is the *debug screen*. This screen, available from various places in the GUI, allows the developer to see the exact content of some specific objects. It lets the programmer see the values of the members of this instance (even private ones) as well as the referenced objects and their content. An example of this *debug screen* is shown in Figure B.6, where the details of a planned CIWS fire action is given.



```
$$$  GunFireAction  $$$  @1679a2
    (int)                             _nbSalvos                1
    (int)                             _roundsInSalvo           55
    (int)                             _stirLocation            -8
    (int)                             KILL_ASSESS_ALL          3
    (int)                             KILL_ASSESS_NONE         2
    (int)                             KILL_ASSESS_ON_OBJ_DEATH 0
    (int)                             KILL_ASSESS_ON_TIME_EXP  1
    (int)                             _killAssessFlag          3
    (float)                           _killAssessTime          58.194466
    (int)                             _targetId                4
    (PositionData)                    _intercept               @3170cc
        (float)                       _positionX               154123.4
        (float)                       _positionY               96966.09
        (float)                       _positionZ               233.31119
    (int)                             _type                    7
    (int)                             PLAN_NONE                -1
    (float)                           _maxTime                 60.135635
    (int)                             _planId                  -1
    (float)                           _plannedStartTime        55.11113
    (int)                             _priority                1
    (float)                           _realStartTime           -1.0
    (boolean)                         _stillPossible           true
    (Vector)                          _conditionList           size:6  @840e0ble
        (ResourceTargetsCondition)    @389cc9
        (ResourceStatusCondition)     @bd5d91
        (ResourcesLeftCondition)      @4a198b
        (ObjectsExistanceCondition)   @d30850
        (TargetInDirectionCondition)  @c6e290
        (TargetInElevationCondition)  @1448fa0
```

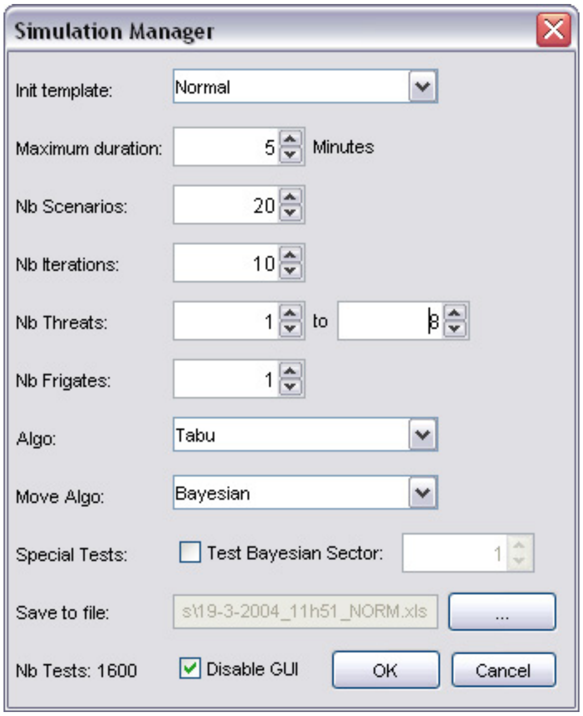
Figure B.6: *Debug screen*

In this screen are given the planned time of execution (at 55.11113*sec*), the hard deadline (at 60.135635*sec*), the list of preconditions that must be met before firing, etc.

B.4 Simulation control

When starting the simulator, the file `tests.cfg` is loaded, if it exits. In this file there is a flag used to enable or disable automated testing. If automated testing is enabled, the remainder of the configuration file is used to set the testing environment. If this flag is disabled, the configuration file is left unused and the simulator GUI is launched as usual.

The following are the parameters that permit tailoring the situations to be tested. The parameters marked with an asterisk ($*$) are shown in Figure B.7 of the simulation manager template.



Simulation Manager

Init template: Normal

Maximum duration: 5 Minutes

Nb Scenarios: 20

Nb Iterations: 10

Nb Threats: 1 to 8

Nb Frigates: 1

Algo: Tabu

Move Algo: Bayesian

Special Tests: ☐ Test Bayesian Sector: 1

Save to file: s\19-3-2004_11h51_NORM.xls

Nb Tests: 1600

☑ Disable GUI

Figure B.7: *Simulation manager*

Parameter	**Description**
*Maximum duration**	Controls the maximum duration of single scenarios. Even if the scenario is not over (there are still ASMs or airplanes with ASMs left in the simulation) when the maximum time specified is elapsed, the scenario is ended.
*Number of scenarios**	This is the number of scenarios executed for each combination of parameters (planning algorithm, movement algorithm, coordination mechanism, threat number, etc.).
*Number of iterations**	This is the number of iterations performed for each scenario. This means that, for each combination of parameters, the number of tests to be made will be (*Number of scenarios*×*Number of iterations*).
*Min/Max number of threats**	These are the maximum and the minimum numbers of threats that will be present in a scenario. If these two numbers are set to be different, all possible values between the two numbers will be used during scenario generation.

*Planning algorithm**	This is the algorithm used for the planning in the tests. The user can choose either 1) to use a single algorithm or 2) to test with all algorithms. In the case where all algorithms are tested, one combination will be generated for each algorithm available.
*Movement algorithm**	This is the algorithm used for movements in the tests. The user can choose either 1) to use a single algorithm or 2) to test with every algorithm. In the case where every algorithm is tested, one combination will be generated for each algorithm available.
Coordination mechanism	This is the mechanism used for multi-agent coordination in the tests. The user can choose either 1) to use a single mechanism or 2) to test with all mechanisms. In the case where all mechanisms are tested, one combination will be generated for each mechanism available.
Formation	This is the ship formation to test, defining the relative position of each platform in the Task Group. This parameter is used only when a coordination mechanism is tested. A combination will be generated for each formation to test.
Distance	The distance between the ships in a coordination formation. Usually, it represents the distance to the center (e.g., the protected High Value Unit -HVU- such as the cargo ship) along one axis. This parameter is used only when a coordination mechanism is tested.
Communication preparation delay	It represents the time to correctly prepare a message with security measures and the correct headers. This is invariant and independent of the size of the messages. This parameter is used only when a coordination mechanism is tested.
Bandwidth	This is the bandwidth of the communication channel, and it is fixed for the length of the simulation. Thus, the bandwidth can be reduced to represent background noise or degraded communication conditions. This parameter is used when any coordination mechanism is tested.
Communication waiting time	This represents the time any agent has to deliberate and return a response. When waiting for a reply, an agent will wait a specific time defined by: *Time to send the initial message + Communication waiting time + Estimated time to receive the reply.* This parameter is only used when a coordination mechanism is tested.

Frigate per threat	This is the number of frigates that will engage each incoming threat. The values of this parameter range from one frigate per threat to all frigates for each threat. This parameter is used only when coordinating with the Contract Net protocol.
Allocation algorithm	Some coordination mechanisms (*Central coordination* and $\sim$*Brown coordination*) compute a matrix of success probability, which contains the evaluation of probability to destroy each threat, for each frigate. When this matrix is obtained, two allocation algorithms can be used: a complete state lookup and a greedy algorithm.
Maximum ship weight deviation	The $\sim$*Brown coordination* uses different ship weighting related to each ship's importance. Once obtained, the weights are normalized in such a way that the maximum weight is 1 and the minimum is $(1-$*Ship weight deviation*$)$.
Fleet engagement priority evaluation	In the $\sim$*Brown* method, the priority of each threat according to the fleet is evaluated from the received probabilities of success (P_S) for each frigate. These fleet priorities will later be used in the evaluation of the individual priority. There are three different fleet engagement priority evaluations: the mean of P_S, the highest P_S and the multiplication of P_S.
Capability matrix evaluation	In the $\sim$*Brown* mechanism, a capability matrix is computed by each ship at a certain moment. Many different evaluation methods can be tested.
Backup	This parameter represents whether or not ships will demand backup in the case where they cannot engage a threat with a sufficient probability of success. This is used only in the *Zone Defence* coordination mechanism.
Threshold	Used in the *Zone Defence* coordination method, this represents the probability of success threshold under which a ship will seek assistance in the engagement of a threat.
*Number of frigates**	Used only in the *Area Defence* coordination mechanism, this is the number of frigates in the scenarios to test. It is used to evaluate the effects of more or less defending ships on an AAW scenario. In the other coordination protocols, the defined formation is used with exactly four frigates.

*Bayesian sector**	This is the Bayesian sector to be tested. Further details about Bayesian sectors are available in [18] and [19]. This restricts the random appearance of threats in a specific sector (based on the ship positioned in the center of the simulation area). This is a special test and is used only to generate the results of the *Bayesian movement* approach.
*Output**	This is the file where the outputs are saved. Typically, they are saved in Excel format (.xls), though results can also be saved in comma separated values format (.csv).

B.5 Choice of development tool

A study was performed to choose an agent development tool for this project. A tool that would help construct a Multi-Agent System (MAS) is required. This study was based on the following requirements for the tool:

1. the best documentation,
2. the best support,
3. the best "quality versus price",
4. the best interface to other languages,
5. the best learning environment.

Such a tool should also deal with aspects more specific to the application at hand such as:

1. multi-agent planning,
2. event management,
3. parallelism,
4. teamwork.

During the study, the following tools for the development of Multi-Agent Systems (MAS) were compared.

1. **OAA** (see Table B.4) – The complete name of the tool is Open Agent Architecture (OAA). This is a free tool made by SRI International. OAA is a framework for integrating a community of heterogeneous software agents in a distributed environment. It provides a facilitator agent that connects all other agents of the system. So if an agent wants to communicate with another one, it has to send a message to the facilitator that will redirect this message to the other agent. All communications between agents use a language called Inter-agent Communication Language (ICL).

Criteria	Evaluation & Remarks
Documentation	Very good
Technical support	None
Reactive/deliberative	Communications have to pass through the facilitator so they could be very time consuming
Interface other languages	Poor: it uses its own communication language (ICL)
Prototyping	Not adapted for this
Development facilities	Medium: ICL seems quite complex at first sight
Learning facilities	Medium: ICL has to be leaned
Current projects	35 applications
Price	Free
Teamwork	No specific module but OAA help the communication between agents
Subjective remarks	This program seems well made. ICL is not easy to learn especially for those who do not know Prolog. This tool seems more specialized in communication between agents than in agent programming in general

Table B.4: *Evaluation of Open Agent Architecture (OAA)*

2. **AgentBuilder** (see Table B.5) – This is a commercial tool from Reticular Systems Inc. It has a toolkit environment to help the development of a Multi-Agent System (MAS) and a run-time environment system in which the agents are executed. Communications are in Knowledge Query and Manipulation Language (KQML). It is implemented in Java and the agents are also in Java, so they can be executed on every operating system that supports Java. The price for this tool was $\$US5000$.

3. **Zeus** (See Table B.6) – This is a free tool developed by British Telecommunications. Zeus provides a library of software components and tools that facilitate the rapid design, development and deployment of agent systems. Communications are in FIPA-ACL. It is implemented in Java.

4. **JACK Intelligent Agents** (see Table B.7) – This is a tool made by Agent Oriented Software (AOS). It uses the JACK^{TM}Agent Language that is a programming language that extends Java with agent-oriented concepts: Agents, Capabilities, Events, Plans, Agent Knowledge Bases (Databases), Resource and Concurrency Management. There is no specific communication language. The price for this tool was $\$US600$.

Taking into account all these aspects, JACK^{TM}Intelligent Agents[18] was selected, because it was the tool that corresponded best to the project needs. First it has good documentation and support. Second, it allows traditional Java programming. Only simple additions to Java have to be learned. Also, JACK^{TM}is not a constraining tool; it offers a great flexibility. The communications between agents is well managed, so time does not need to be spent on it. Another good thing about JACK^{TM}is that it has a good plan choice system that

---

[18] See http://www.agent-software.com

Criteria	Evaluation & Remarks
Documentation	Good, the documentation seems quite complete
Technical support	Excellent, professional technical support with mailing list
Reactive/deliberative	Mix of BDI and rule-based
Interface other languages	Bad, agents have to be executed in the AgentBuilder environment
Prototyping	Excellent, easy and fast development
Development facilities	Excellent, easy to use and very visual
Learning facilities	Good, the learning is relatively simple because of the good documentation and the development toolkit
Current projects	A lot like NASA, US Army, Boeing, Aerospace, etc
Price	Expensive, $\$US4995$
Teamwork	No specific module, but it uses the KQML language to communicate between agents
Subjective remarks	Very good tool, but quite expensive and separate from traditional programming. We have to use their tools to program

Table B.5: *Evaluation of AgentBuilder*

helps with the planning system. And finally, this tool has already been used in defence and aerospace applications.

B.5.1 Description of JACK intelligent agents

JACK$^{\mathrm{TM}}$Intelligent Agents is an Agent Oriented development environment built on top of and fully integrated with the Java programming language. It includes all components of the Java development environment as well as it offers specific extensions to implement agent behavior. JACK$^{\mathrm{TM}}$relationship to Java is analogous to the relationship between the C++ and C languages. C was developed as a procedural language and subsequently C++ was developed to provide programmers with object-oriented extensions to the existing language. Similarly, JACK$^{\mathrm{TM}}$has been developed to provide agent-oriented extensions to the Java programming language. JACK$^{\mathrm{TM}}$source code is first compiled into regular Java code before being executed.

The agents used in JACK$^{\mathrm{TM}}$are intelligent agents. They model reasoning behavior according to the theoretical Belief, Desire, Intention (BDI) model of artificial intelligence. Following the BDI model, JACK$^{\mathrm{TM}}$intelligent agents are autonomous software components that have explicit goals to achieve or events to handle (desires). To describe how they should go about achieving those desires, these agents are programmed with a set of plans. Each plan describes how to achieve a goal under varying circumstances. Set to work, the agent pursues its given goals (desires), adopting the appropriate plans (intentions) according to its current set of data (beliefs) about the state of the world. This combination of desires and beliefs initiating context-sensitive intended behavior is part of what characterizes a

Criteria	Evaluation & Remarks
Documentation	Good : it discusses both modeling and how to use the tool
Technical support	Nothing formal
Reactive/deliberative	Rule-based
Interface other languages	The java code is automatically generated so it could be more difficult to use in other applications
Prototyping	Good: it seems easy and fast to program
Development facilities	Good: easy to use but not as easy as AgentBuilder
Learning facilities	Good: due to the graphic interface and the good documentation
Current projects	Information not available
Price	Free
Teamwork	No specific module
Subjective remarks	The automatically generated code makes the development easier but it is not known exactly what is generated. This could make the use of other architectures more difficult.

Table B.6: *Evaluation of Zeus*

BDI agent.

A JACK^{TM}agent is a software component that can exhibit reasoning behavior under both pro-active (goal directed) and reactive (event driven) stimuli. Each agent has:

1. a set of beliefs about the world (its data set),
2. a set of events that it will respond to,
3. a set of goals it may desire to achieve (either at the request of an external agent, as a consequence of an event, or when one or more of its beliefs change), and
4. a set of plans that describe how it can handle the goals or events that may arise.

The JACK's components are essentially the following ones

1. The **JACK Agent Language** — The JACK Agent Language is the actual programming language used to describe an agent-oriented software system. The JACK Agent Language is a super-set of Java, encompassing the full Java syntax while extending it with constructs to represent agent-oriented features.
2. The **JACK Agent Compiler** — The JACK Agent Compiler pre-processes JACK Agent Language source files and converts them into pure Java. This Java source code can then be compiled into Java virtual machine code to run on the target system.
3. The **JACK Agent Kernel** — The JACK Agent Kernel is the runtime engine for programs written in the JACK Agent Language. It provides a set of classes that give JACK Agent Language programs their agent-oriented functionality.

Criteria	Evaluation & Remarks
Documentation	Good: the documentation seems quite complete
Technical support	They have an e-mail address and they respond very fast and with good answers
Reactive/deliberative	Agents can exhibit reasoning behavior under both pro-active (goal directed) and reactive (event driven) stimuli
Interface other languages	The JACK^{TM}Agent Language is compiled in Java. It supports Java 100%.
Prototyping	Medium: the conception is accelerated with JACK^{TM}but it is not as simple as other tools.
Development facilities	Traditional programming in Java.
Learning facilities	Medium: the JACK^{TM}Agent Language has to be learned.
Current projects	Many: like "Cable & Wireless" – "Optus" – "Alcatel" and defence organisms in Australia and Canada
Price	$\$US600$
Teamwork	It includes a teamwork module but it seems quite constraining
Subjective remarks	It seems quite strong and relatively close from standard programming (there is no wizard that automatically generates code). Learning the JACK^{TM}Agent Language makes it more complex and long to learn how to use the tool.

Table B.7: *Evaluation of JACK^{TM}Intelligent Agents*

In summary, the JACK Agent Language is closely related to Java and extends the regular Java syntax. It allows the programmers to develop the components that are necessary to define BDI agents and their behavior. These functional units are:

1. **Agents** – They have methods and data members just like objects, but also contain capabilities that an agent has, database relations that they can use to store beliefs, descriptions of events that they can handle and plans that they can use to handle them.

2. **Capabilities** – They serve to encapsulate and aggregate functional components of JACK Agent Language for use by agents. Capabilities can include events, plans, databases or even other capabilities.

3. **Database Relations** – They are used to store beliefs and data that the agent has acquired. Agents can also use regular Java data structures for storing information, but the advantage of a database is that it will generate events when particular changes are made.

4. **Views** – They provide events when particular changes are made.

5. **Events** – They identify the circumstances and messages that it can respond to.
6. **Plans** – They are executed in response to these events.

Each event, plan, and database used are implemented as Java classes. They inherit certain fundamental properties from a base class and then extend these base classes to meet their own specific needs. The base classes are defined within the kernel and form the "glue" that holds a JACK agent-oriented program together. However, the JACK Agent Language is more than just a specific organization of Java objects and inheritance structures; it provides its own extended syntax, which has no analogous representation in Java.

B.5.2 Tools for adapting JACK agents to C++

To ultimately port Jack agents to more powerful test bed environments (such as the Ship Air Defence Model of BAE), it is desirable to be able to adapt these agents to be compatible with C++. An investigation was performed on the advantages and disadvantages of various tools that allow the use of JACK agents in C++.

The tools needed must provide a bridge between Java and C++ data structures. They must also be compatible with a variety of host platforms. A survey of tools suggested the following as the most promising options.

CORBA

Advantages

- Easy to use method invocations and data structures in both languages.
- Brings utilities of its own (list of objects) that both languages can use.
- Safe for both applications Java Virtual Machine (JVM)
  - Enables catch and throw of errors.
- Can be used with other languages, *e.g.*, Cobol, Small Talk, ADA.

Disadvantages

- Mostly useful for network communication applications.
  - Heavy solution for "local" use.
- Shared classes need to be declared in the IDL file.
- Cannot share classes that are not declared in IDL.
  - Lose Java's useful packages.
  - Lose C++ specific classes, STL, etc.

JNI

Advantages

- Brings the entire JVM to the C++ code.
  - Receive Java's thread pointers.
  - Attach your C++ thread to the JVM.
  - Allows you to use any Java packages.
- Native (C++) method is part of a Java Class and can then access any of its members.
- Easy to implement in your Java code.

Disadvantages

- Lots of work for the C++ bridge code.
  - Any method called needs to be bound on the JVM before you can use them.
  - Any Java object used needs to be bound too.
- Programmers who code the C++ bridge need to be super careful.
  - Always watch for errors thrown on the Java side.
  - Could damage the usually secure JVM.

JACOB

Advantages

- Secure, since it only transfers data.
- Allows you to write data in XML, ASCII, binary.
- Lets you use your data any way you want.
  - Includes read and write methods for shared classes.
- Could be useful for database transfer.
- Part of the developed agent suite (JACK).

Disadvantages

- Does not bring much in the way of utilities.
  - It is mostly just serialized classes.
- Not much support or knowledge bases from AOS.
  - Mainly for C++ use.
- Not designed to be used (freely) in C++.

- Shared data structures need to be declared in an XML file.

Figures B.8 and B.10 give a comparison of the different approaches. It can be noticed that CORBA is easy to use and very portable, but it is slow and heavy. JNI is rapid, made for local calls, supports the JVM in C++, and is backed by Sun. However, it is not easy to use, and it must be taken with care. JACOB is easy and reliable, but mainly because it does not offer much. In addition, there is not much support for it. For the application targeted by this project, JNI was found faster and offers many capabilities for development. It has much fewer boundaries and is worth the difficult coding. For less complex applications, the CORBA might be the choice. It provides secure and reliable code that can be used by different language-based applications within a network.

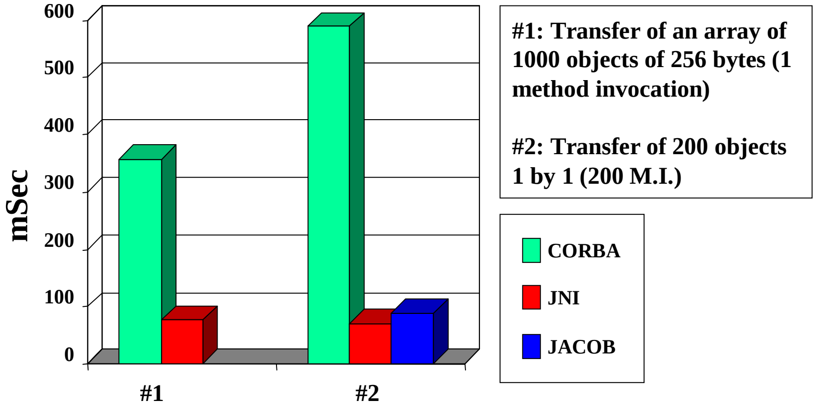


Figure B.8: *Comparison of data transfers with CORBA, JNI, and JACOB*

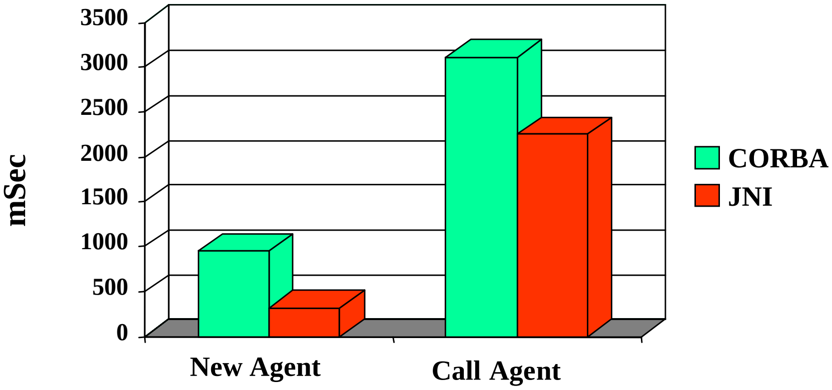
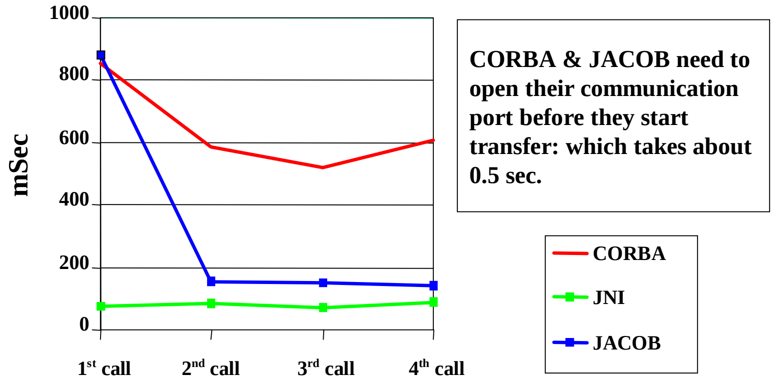


Figure B.9: *Comparison of agent usage with CORBA and JNI*



CORBA & JACOB need to open their communication port before they start transfer: which takes about 0.5 sec.

Figure B.10: *Comparison of calls in time with CORBA, JNI, and JACOB*

This page intentionally left blank.

Annex C: Assumptions

C.1 Description of threats

1. The scenario has at most 8 threats.
2. All threats exist at the beginning of the scenario (*i.e.*, no new threats appear after the initial conditions are set).
3. Assume all threats are a single type of anti-ship missile.
4. Threats are all incoming directly toward ownship position.
5. Threats travel in a straight line.
6. Speed constant at Mach 2.5.
7. Threats are generated randomly at the range R from ownship: $5km < R < 80km$.
8. Threats are generated randomly at any azimuthal angle about ownship.
9. Threats are generated randomly at the polar angle ϕ: $0° < \phi < 90°$.
10. The probability of a threat that reaches ownship causing a kill is 50%.
11. The specification of the absolute threat rating is inversely proportional to the "time of flight" of the threat to ownship.
12. All kinematic information is known in three dimensions (*i.e.*, not restricted to range and bearing data only).

C.2 Doctrines

1. Both Separate Tracking and Illumination Radars(STIRs) will not be simultaneously assigned to the same threat.
2. Multiple weapon resources (SAMs, gun) will not be controlled by the same STIR.
3. Neither a SAM nor the gun will engage or re-engage a threat until KA of a prior engagement is complete.
4. A threat will not be engaged or re-engaged with a SAM or the gun if it is engaged with the CIWS.
5. Note that nothing will preclude the Close-In Weapon System (CIWS) from engaging a threat simultaneously with a SAM or the gun if one of the latter two were engaged first.
6. In this version of the problem, it is assumed that, once engaged, the CIWS will fire at a single threat until the threat is killed or until the threat has reached ownship.

7. The CIWS will work in an autonomous mode of operation which independently searches for and engages targets that satisfy some pre-determined engagement criteria. Note that there is another mode of operation in which, if a STIR is locked on a target, the CIWS can be given the precise position to begin its search and lock-on. The time to do so is negligible, and it leads to directed firing on a specific target.

Distribution list

DRDC Valcartier TR 2005–486

Internal distribution

1	Director General
3	Document Library
1	Head/Decision Support Systems (author)
1	M. Bélanger
1	Dr P. Valin
1	Head/Information and Knowledge Management
1	R/Visualization & Geo-spatial Systems
1	R/Knowledge Management Systems
1	R/Informations System Warfare
1	Head/System of Systems
1	R/Metrics & Experimentation
1	R/Distributed Synthetic Environment
1	R/System Engineering & Architecture
1	LtCol. B. Carrier
1	LCdr É. Tremblay
1	Maj. B. Deschênes
1	Dr M. Allouche
1	Dr A. Benaskeur (author)
1	J. Berger
1	Dr R. Breton
1	M. Blanchette
1	Dr A. Boukhtouta
1	Dr A.-C. Bourry-Brisset
1	C. Daigle

1	Dr A. Guitouni
1	Dr H. Irandoust
1	Dr A.-L. Jousselme
1	P. Maupin
1	S. Paradis
1	F. Rhéaume
1	A. Sahi
1	J.M.J. Roy

Total internal copies: 34

External distribution

1	DRD KIM (PDF file)
2	Director Science & Technology Maritime(DSTM)
2	Director Science & Technology Land (DSTL)
2	Director Science & Technology Air (DSTA)
2	Director Science & Technology C4ISR (DSTC4ISR)
1	Director Maritime Requirements Sea (DMRS) 4
1	Director Maritime Requirements Sea (DMRS) 6
1	Director Aerospace Requirements (DAR) 4
1	Director Aerospace Requirements (DAR) 4-2
2	Director Maritime Ship Support (DMSS) 6
2	Director Maritime Ship Support (DMSS) 8
2	DRDC - Atlantic: Attn: Dr. Bruce MacArthur Attn: Dr. Jim S. Kennedy
2	DRDC - Ottawa: Attn: Barbara Ford Attn: Dan Brookes
2	CF Maritime Warfare School CFB Halifax Halifax, Nova Scotia Attn: TAC AAW OIC Modeling and Simulation
2	Canadian Forces Naval Operations School CFB Halifax Halifax, Nova Scotia Attn: Tactic CT AWW
1	Canadian Forces Naval Engineering School CFB Halifax Halifax, Nova Scotia Attn: CSTC
1	Operational Requirements Analysis Cell CFB Halifax Halifax, Nova Scotia Attn: Commanding Officer

1 Canadian Forces Fleet School CFB Esquimalt
Esquimalt, British Columbia
Attn: Commanding Officer/WTD

1 Operational Requirements Analysis Cell CFB Esquimalt
Esquimalt, British Columbia
Attn: Commanding Officer

Total external copies: 29

Total copies: 63

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence R&D Canada – Valcartier 2459 Pie-XI Blvd. North, Québec, Québec, Canada G3J 1X5	2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable). UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title).

Combat resource allocation planning in naval engagements

4. AUTHORS (last name, first name, middle initial)

Benaskeur, A.; Bossé, É.; Blodgett, D.

5. DATE OF PUBLICATION (month and year of publication of document) August 2007	6a. NO. OF PAGES (total containing information. Include Annexes, Appendices, etc). 100	6b. NO. OF REFS (total cited in document) 19

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered).

Technical Report

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include address).

Defence R&D Canada – Valcartier
2459 Pie-XI Blvd. North, Québec, Québec, Canada G3J 1X5

9a. PROJECT NO. (the applicable research and development project number under which the document was written. Specify whether project). 11bm	9b. GRANT OR CONTRACT NO. (if appropriate, the applicable number under which the document was written).
10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique.) DRDC Valcartier TR 2005–486	10b. OTHER DOCUMENT NOs. (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution beyond the audience specified in (11) is possible, a wider announcement audience may be selected).

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

Reaction times of modern current and future war platforms are eroded, since they are expected to operate in a large variety of complex scenarios. To cope with the increasingly diverse air and surface threats, modern platforms, either operating in a single ship configuration or within a task group, will require their sensor suite and weapon arsenal to be efficiently managed. The coordination and tight integration of these resources will also be required.

The Decision Support Systems (DSS) Section, at Defence Research & Development Canada – Valcartier (DRDC Valcartier), has initiated collaboration with industry and university partners. This collaboration aims at developing and demonstrating advanced concepts of combat resource management. The latter could apply to the current Command & Control Systems (CCSs) of the Halifax and Iroquois Class ships, as well as their possible future upgrade (*i.e.*, SCSC platform), in order to improve their performance against predicted future threats. This activity builds upon and broadens the scope of prior research in the domain. It is oriented to the study, development, and implementation of management decision aids for tactical shipboard resources, based on intelligent agent technology and techniques for multi-agent planning and coordination.

This report presents a review of agent and multi-agent planning approaches. Theoretical basis of agent and multi-agent systems are introduced and planning problems are described. The results of the implementation and test of different algorithms for hardkill and softkill combat resource allocation, in naval engagements, are presented and discussed.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title).

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

DEFENCE DÉFENSE

WWW.drdc-rddc.gc.ca