Control Architecture for Multiple Autonomous Unmanned Vehicle Operations - Research Proposal

F. Maurelli, J. Cartwright, and D. M. Lane
Ocean Systems Laboratory
Department of Electrical, Electronic and Computer Engineering
School of Engineering and Physical Sciences
Heriot-Watt University

DRDC CORA CR 2012–100
May 2012

Defence R&D Canada
Centre for Operational Research and Analysis

Maritime Operational Research Team
Director of Maritime Strategy

National Defence Défense nationale

Canada

Control Architecture for Multiple Autonomous Unmanned Vehicle Operations

Research Proposal

F. Maurelli
J. Cartwright
D. M. Lane
Heriot-Watt University Ocean Systems Laboratory

Prepared by:

Ocean Systems Lab
Department of Electrical, Electronic and Computer Engineering
School of Engineering and Physical Sciences
Earl Mountbatten Building
Heriot-Watt University
Riccarton, Edinburgh, EH14 4AS Scotland

Project Manager: D. M. Lane, +44 (0) 131 451 3328
Contract Number: W7714-115079/001/SV
Contract Scientific Authority: B. U. Nguyen, 613-947-9759 and F-.A. Bourque, 613-992-3206

Defence R&D Canada – CORA

Contract Report
DRDC CORA CR 2012-100
May 2012

Principal Author

Original signed by F. Maurelli

F. Maurelli

Approved by

Original signed by P. L. Massel

P. L. Massel
Acting Section Head Maritime and ISR Systems OR

Approved for release by

Original signed by Paul Comeau

Paul Comeau
Chief Scientist

Abstract

This report forms the first deliverable of the DRDC - Control Architecture for Multiple Autonomous Unmanned Vehicle Operations, under Contract #W7714-115079/001/SV. This project addresses the topic of a decentralised world model service that operates across multiple underwater vehicles. After an outline of the related literature, the report focuses on the open challenges and on the proposed approach. The main features are an adaptive mission planning system, interfaced with a World Model, based on Ontology representation. An important focus is given to the communication among vehicles, considering a temporal context and the restrictions of the underwater domain.

Résumé

Ce rapport constitue le premier résultat de la structure de commandement de multiples véhicules autonomes sans pilote de RDDC, conformément au contrat no W7714-115079/001/SV. Ce projet porte sur le système décentralisé d'un modèle du monde pouvant commander plusieurs véhicules sous marins. En plus d'un aperu de la documentation connexe, le rapport met l'accent sur les défis actuels et l'approche proposée. Les principales caractéristiques comprennent un systme de planification adaptative de missions lié un modèle du monde, en fonction de la représentation de lontologie. La communication entre véhicules tout en tenant compte du contexte temporel et des contraintes du domaine sous-marin constitue un important point de mire du projet.

This page intentionally left blank.

Executive summary

Control Architecture for Multiple Autonomous Unmanned Vehicle Operations

F. Maurelli, J. Cartwright, D. M. Lane; DRDC CORA CR 2012-100; Defence R&D Canada – CORA; May 2012.

Introduction: AUV missions are increasing and cooperation among a fleet of AUVs is a need. This project addresses the topic of a decentralised world model service that operates across multiple underwater vehicles.

Literature Review: A selection of the literature is presented. For clarity of presentations, three blocks have been identified: Multi-robot Cooperation, Communication, Knowledge Representation and Ontologies.

Challenges: This section presents the challenges which are still open and that this project aims to address.

Proposed System: The proposed system is based on an adaptive mission planning system, communicating with a World Model system, which represents and updates the model of the world, based on pre-acquired knowledge and updates from sensors. A particular emphasis is given to the communication among vehicles, considering a temporal context and the restrictions of the underwater domain.

Sommaire

Control Architecture for Multiple Autonomous Unmanned Vehicle Operations

F. Maurelli, J. Cartwright, D. M. Lane ; DRDC CORA CR 2012-100 ; R & D pour la défense Canada – CARO ; mai 2012.

Introduction : Les véhicules sous-marins autonomes (AUV) sont de plus en plus sollicités lors de missions. La coopération au sein d'une flotte est ainsi nécessaire. Ce projet porte sur le système décentralisé d'un modèle du monde pouvant commander plusieurs véhicules sous-marins.

Analyse documentaire : De la documentation est remise. Afin de clarifier les présentations, les trois éléments suivants ont été identifiés : coopération ; multi robots ; communication ; représentation des connaissances et ontologies.

Défis : Cette section présente les défis actuels qu'adressera le projet.

Structure proposée : La structure proposée se base sur un système de planification adaptative de missions communiquant avec le système dun modèle du monde mis à jour en fonction des connaissances acquises et des données fournies par les capteurs. La communication entre véhicules tout en tenant compte du contexte temporel et des contraintes du domaine sous-marin constitue un important point de mire du projet.

Table of contents

List of figures

This page intentionally left blank.

1 Introduction

The coming generation of Autonomous Underwater Vehicle (AUV) missions are requiring that vehicles operate without human assistance for extended periods of time (days), in environments that are imprecisely known. Examples of such persistent autonomy needs are present in over the horizon surveillance or mapping (for security and marine science), and in deep-water oilfields (for inspection, repair and maintenance). To accelerate progress of these missions, multiple vehicles may be expected to operate simultaneously, exhibiting collaboration in their execution of tasks.

Equipping vehicles to execute these missions successfully requires that they are able to adapt their mission plans dynamically in real-time, using mechanisms such as plan repair or interleaving planning and execution. Crucial to such online mission adaptation is a representation of the environment or world that can be used as a basis for selecting future actions. The world model typically includes not only the physical state of objects in the environment (including collaborating platforms), but also a degree of internal state, such as the health and intentions of other platforms. An example of world modelling for AUVs is given in [1]. There, they store information in semantic form using ontologies [2], which support high level deliberative vehicle control. However, theirs is a single vehicle system.

This project addresses the topic of a decentralised world model service that operates across multiple underwater vehicles, sitting above the communications data link layer, with a multi-vehicle adaptive planner which is able to schedule and execute actions with the best ratio reward to cost. Figure 1 illustrates how the world model fits in a multi-vehicle system. It presents an abstraction away from data packets and acknowledgements, providing each vehicle with a partially shared world view at the semantic information level. This removes the need for vehicle control systems to explicitly plan speech acts, whilst giving the world model the freedom to optimise information exchange across the group of vehicles. The decentralised nature of the system ensures that it is robust to temporary node isolation. We target the most capable underwater communications mechanism, acoustic modems.

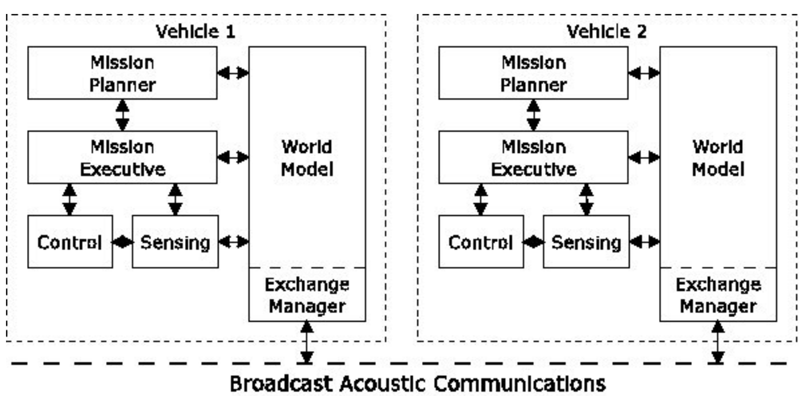

Figure 1: *The world model in a simple three-layer robot control architecture*

2 Literature Review

2.1 Multi-robot cooperation

In this section, we take a broad look at the history and current state of the art of cooperative autonomous robotics. Whilst the field contains a large body of literature, the realisation of generic, robust cooperative systems is still some way off.

Early autonomous robotics research was severely restricted by the computational limitations of the hardware present at the time. Early theoretical research into cooperative multi robot systems was made by Lesser and Corkill [3], but no apparent implementation or trials were attempted.

Albus and Blidberg presented a control system architecture for multiple autonomous undersea vehicles (MAUV), complete with a world model representation, as considered by this project [4]. Their system is ambitious, especially for the time; when the paper was published, an implementation was not complete. Given the lack of follow up work presenting evaluation results, the author has to assume that the project was discontinued.

Slightly later in the same decade, the ACTor-based Robots and Equipments Synthetic System (ACTRESS) was developed by Asama *et al.* [5]. Their framework has multiple levels in the message protocol, allowing the exchange of information at multiple levels of abstraction. Whilst focusing primarily on the synthesis of distributed autonomous systems, experiments were conducted using small homogeneous mobile robot platforms ("Micromouse"), controlled by individual personal computers. The task was to move obstacles across a room, and addressed issues of position estimation, obstacle/peer recognition, collision detection, path planning and task assignment. This is one of the first examples of multi-robot cooperation using real robots.

The ALLIANCE architecture by Parker [6] was designed to allow robust, distributed fault tolerant control of cooperative robots. A further publication appears to have originated from Parker's thesis, four years later [7]. Robots in the ALLIANCE architecture are embodied with motivations such as impatience and acquiescence, and an adaptive action selection mechanism is used. In the experimental environment, three robots cooperated in a simplified hazard waste cleanup task. Whilst effective in this situation, the author believes an architecture with deliberative planning is more applicable to a wide range of tasks.

Jung *et al.* present experiments with cooperative cleaning robots "Flo" and "Jo", measuring their performance at different levels of cooperation, from emergent cooperation with no communication to explicit cooperation and communication [8]. They showed that on this task, explicit communication allowed the most efficient cooperation. However, the communication involved was very basic, restricted to object positions.

The problem of collaborative multi-robot exploration is tackled by Burgard *et al.* [9]. Their system is based on a shared occupancy grid map, and was shown to work in both simulation and real trials. No attempt is made to model the world other than for mapping, however.

The literature contains a number of papers on multi-robot systems for RoboCup robot soccer competitions, both simulated and physically embodied. Examples include Candea *et al.* [10], Bruce *et al.* [11] and Roth *et al.* [12]. As stated in [10], the RoboCup environment is very dynamic, and the opposing team provides an additional challenge unseen in most multi-robot tasks in the literature. However, the competitive aspect of the games means that specialised architectures are typically favoured over more flexible generic ones, due to performance pressures.

From the literature reviewed above, it appears that whilst several system have been produced for collaborative mobile robot operations, those that have actually reached the implementation and testing stages are significantly limited in scope. Particularly in the sense of world representation and communication, the solutions are often closely tied to the application. With a view to more generic solutions employing structured world models, the next section considers literature from the field of agent communication.

2.2 Communication

In writing this section, reference was made to a comprehensive review on agent communication by Labrou [13].

One early attempt at standardising communication for knowledge sharing was the Knowledge Query and Manipulation Language (KQML) [14], a product of the DARPA-sponsored Knowledge Sharing Effort (KSE), developed by Finin *et al.*. It was not initially intended for use as an agent communication language (ACL), but was later put to that purpose. KQML is a high-level message-oriented communication language and protocol for exchange of arbitrary information, independent of the information content. A possible syntax for the message context would be the Knowledge Interchange Format (KIF) [15], another output of the KSE.

After researchers in the field gained several years of experience with KQML, and issues were raised with the original version of the language, a revision was created and presented in [16]. Around the same time, the Foundation for Intelligent Physical Agents (FIPA) published their own agent communication standard, FIPA ACL [17], which is (intentionally) similar to KQML in many respects. The latest revision of the FIPA ACL was produced in 2002, and is defined by multiple documents on the website http://www.fipa.org.

As in the revised version of KQML, FIPA ACL has formally defined semantics in the Syntax/Semantic Language (S/SL), described in the ACL specification. These define the preconditions for performing a communication act (e.g. query or inform), and its expected effect. According to the comparison between KQML and FIPA ACL given by [13], there is little to choose between the two languages, and a pragmatic selection should be made.

Rejecting the KQML and FIPA-ACL languages, Walton defines a new ACL called Multi-agent Dialogue Protocols (MAP) [18]. He claims that this "separates the rational process and interactions from the actual dialogue itself".

The authors believe that that whilst potentially very powerful, the above ACLs would add unnecessary complication to this project, given the intended shared semantic knowledge representation that may be used as a reference point. This view seems to be taken by others in the field of multi robot systems, as these languages see little have seen in practical systems such as those reviewed in section 2.1.

2.3 Knowledge Representation and Ontologies

This section reviews literature involving knowledge representation, in the context of world modelling for autonomous mobile robots.

Maio and Rizzi propose a multi-layer architecture for knowledge representation, and describe the progressive abstraction of knowledge from several channels of sensor data available to the agent [19]. Whilst their architecture seems reasonable, they include far too little detail, and no evaluation system is mentioned.

Chandrasekaran *et al.* discuss the use of ontologies for knowledge representation and exchange in a general sense, but their paper is geared more towards intelligent semantic web agents than mobile robots, so many relevant issues are not addressed [20].

The Spatial Semantic Hierarchy (SSH) is described by Kuipers [21], which strives to represent spatial knowledge in different distinct but interacting ways, both quantitative and qualitative. Figure 2 is taken from that paper, and illustrates different representations in two dimensions; the horizontal direction gives the qualitative/quantitative dimension, and in the vertical direction the spatial knowledge representation is organised into levels according to ontology. Whilst interesting, the author considers this framework to be unnecessarily complicated especially given the assumption of metric positional sensors available on the target platforms, e.g. GPS and DVL. GPS does not propagate underwater. DVL sensor is not available on all AUVs. A robust world model should be able to represent the reality without having such information. However, inspiration may be drawn from portions of this work.

Slightly later on, Chella present a representational model for environmental robotic knowledge [22]. Predating modern semantic web ontology languages, they utilise an XML-based markup language for structuring information. They make the assumption that robots acquire information about their environment from three distinct channels: the quantitative

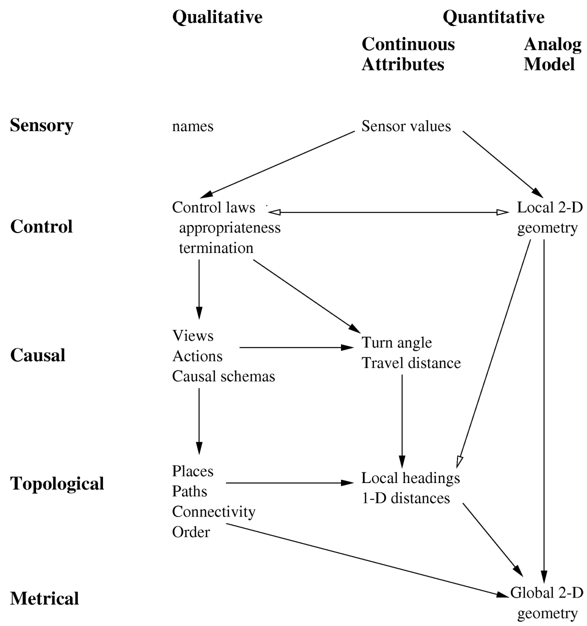

Figure 2: *Diagram showing distinct representations of their Spatial Semantic Hierarchy.*

metric channel, the visual channel, and the semantic channel. Their system revolves around a central agent-based map server that robots can query and update whilst navigating an office environment. However, a centralised architecture is very vulnerable to failure.

In the same year, Hazarika and Cohn publish details of a very formal logic-based framework to construct a qualitative spatio-temporal world model from local surveys [23]. However, they make no attempt to implement or evaluate their theories.

The work of Kokar and Wang [24] appears to be one of the first publications in the literature that utilises a data storage format standard from the knowledge representation community; the DARPA Agent Markup Language (DAML), [25]. The knowledge they store is to assist in target recognition by autonomous systems. Whilst quite specific in scope, this points towards the direction intended for this project.

Another target-related publication by Schlenoff [26] is concerned with reasoning about obstacles in the context of autonomous driving. Specifically, reasoning about the danger posed to the vehicle by obstacles, based on both obstacle and vehicle classification. A later publication Matheus describes how this system was extended to perform vehicle path planning for obstacle collision avoidance [27]. Whilst interesting in terms of representation and reasoning, this work makes no attempt at online world modelling.

Matheus et al. describe an ontological representation of information for battlefield situation awareness [28]. This is geared towards storage on a command and control computer system; they do not consider its use for world models held by individual robots. However, the representations they propose may be useful for this purpose, especially that of temporal context.

A functional world model system with sharing amongst multiple robots is described by Roth *et al.* [12]. In the robot soccer domain (RoboCup), each robot in their system constructs a real time world model and shares portions of this model with its peers. The main limitation of their system is that the model is merely a vector of positions and headings, with no semantic structure.

Some interesting ideas about knowledge representation for robots are given by Balakirsky *et al.* in [29], which presents the output of a workshop at the Knowledge Representation and Ontologies for Autonomous Systems Symposium. The situation given to the attendees was that of collection of rubbish at an airport by mobile robots. The paper highlights the range of possible knowledge representations for a given task, offering different advantages and disadvantages. It also confirms the difficulty of the knowledge representation problem for autonomous systems, suggesting that it should receive more attention from the community.

The benefits of using ontologies for level-one sensor fusion (data related to detection, tracking, classification and identification), are touted by Ceruti [30]. One of his main points is the potential to perform inference over the data stored in a suitable ontology, such as highlighting the possible threat posed by a particular ship travelling at a given speed within a certain area. Unfortunately they do not go further, with no attempt made at implementation or evaluation of such a system.

Kim *et al.* use ontologies to provide contextual information for a mobile robot in an office environment [31]. They model three ontologies: an environment ontology to model office rooms, their connectivity and content; an object ontology to hierarchically model features of objects such as chairs and tables; and a task ontology, giving tasks with goals, preconditions, actions and effects (hierarchical task decomposition is used). Unfortunately little further detail is given. They state that they have implemented the system they describe, but don't give any results for its evaluation, either qualitative or otherwise.

Ontological description of a mobile robot is taken to a finer level of detail by Amigoni and Neri [32], who model robots down to their individual wheels. They use this model for control purposes, with the ability to test if a certain plan of movement is achievable given a robot's locomotion system, and its dimensions and obstacles it must avoid. Whilst an example of the vehicle ontology is presented, they do not give details of the external world representation.

An Ontology Based Object Categorisation system (OBOC) for Sony AIBO RoboCup agents is described by Mendoza and Williams [33]. Their system aims to perform feature- and context-based classification of observed objects. A proof of concept system was developed on a PC, with the intention that this would be ultimately moved to the AIBO robots. Thus this merely hints at the potential power of ontological technologies for object classification, without evaluating a real system.

A comprehensive ontology was developed for robots operating in the urban search and rescue domain by Schlenoff and Messina [34], but the research covered by that paper had unfortunately not reached the implementation stage. In mention of their future work, they appear to be concentrating on refining and adding to the ontology rather than working towards producing a prototype or proof of concept implementation, which the author would expect. However, Schlenoff does have previous experience of implementing ontological systems [26]. Another detailed ontology was created by Schlenoff for an intelligent ground vehicle [35]. This was an ambitious ontology for representing vehicle knowledge in the context of military manoeuvres, but it is only partially complete, and no implementation was described.

From the literature reviewed in this section, we found that whilst a number of ontologies have been created for autonomous mobile robots, few have actually been implemented, and those that have are very narrow in their scope. Additionally, none of the proposed systems explicitly addresses a temporal framework linked to ontology representation. This project intends to create a generic system for online distributed world modelling on mobile robots.

3 Challenges

The aim of this project is to design and implement a framework for a distributed semantic world model to support cooperation between multiple autonomous vehicles. The system should have the properties given below.

1. Temporal context is incorporated into the world model (lacking in existing systems).
2. Vehicles may exchange information with their peers at different levels of abstraction, significantly increasing on the capabilities of current systems.
3. Selection of information to exchange is performed automatically, with minimal prior input from the vehicle subsystem designers.
4. Robust to communication failure between individual nodes.
5. No central communications point.
6. Vehicles may still operate in isolation, with reference to their local world model.
7. New vehicles and missions can be incorporated in a simple, modular fashion.
8. System design and documentation is sufficiently clear that extension and modification by third parties is possible.

The combination of temporal ontological representation (1) with robustness to communication failure (4,5,6) will be the main novel contribution to the field. The author also considers the last objective (8) to be fundamental to the success of this project. The architecture and code base must be accessible to future researchers if it is to receive continued use.

Evaluation of the framework will necessarily involve creating several ontologies to describe the information representing the vehicles under test and the world they inhabit.

Upon creating a list of goals or requirements for a system, it is equally important to list issues that will *not* be addressed, and any assumptions that will be made. Thus, for this project:

1. Communication will be point to point or broadcast, with no facility to route information to hidden nodes.
2. A media access control mechanism is assumed to be available for any broadcast medium used.
3. The cooperating vehicles share a common purpose, without conflicting motivations.
4. Existing subsystems from the Ocean Systems Laboratory will be integrated to provide robot control, sensing and planning functionality.

The next section presents details of the proposed system.

4 Proposed system

4.1 Control and Planning Architecture

The challenges for providing autonomous mission planning for AUVs were clearly stated by [36], [37], [38]. Approaches previously validated in Space have been able to provide adaptive planning capabilities to oceanographers for maximising the science return of AUV missions. Using learning techniques for the identification of features [39] and deliberative reactors for the concurrent integration of execution and planning, live sensor data can be analysed during mission to adapt the control of the platform in order to measure episodic phenomenon. Other approaches have made use of behaviour-based controllers that generate Pareto-optimal and satisfying behaviours to control the direction and velocity parameters of the host platform [40]. This is a previous work from our lab, which constitutes an important starting point for the work of this project.

Our approach for autonomous decision making is motivated by the need of a service-oriented architecture for multiple assets, a portable and extensible solution, a dynamic and uncertain environment, and the requirement to maximise operability during mission. We propose a novel approach for adaptive mission planning for AUVs operating in a dynamic and uncertain discoverable mission environment. We assume that the information provided by the knowledge base is fully observable to the planner, i.e. the uncertainty arising from sensor limitations is handled by the agents processing lower-level data, e.g. Autonomous Target Recognition (ATR). We also assume that the mission environment is dynamic and uncertain, i.e. external events may occur and actions do not always perform as expected. Under these assumptions, our approach implements a Bayesian paradigm for prediction, measurement, and correction inside a sequential decision-theoretic planning Markov decision process framework. Based on a continuous reassessment of the status of the mission environment, our approach provides a decision making loop capable of adapting mission plans. Instead of solving a plan from initial state to goals like in classical AI planning, it maintains a window of actions that it is believed can be performed from the current state in order to improve a given utility function.

Fig. 3 describes the workflow of the algorithm, providing a high-level overview. At the top of the loop, the PLAN stage computes the best plan candidate π_q from a given mission environment Π_q (or the initial environment Π_0). The first ground action of the plan π_q is instantiated at the EXECUTE stage and executed in the functional layer of the platform via the ground action execution instance e_t^q. This stage produces a new predicted state $\tilde{x}_q$. This state is compared with the one provided by the knowledge base $\dot{x}_q$. Under a normal iteration, the state x_q is corrected based on this comparison at the CORRECT stage and the loop continues. This is the simplest form of the approach. However, a set of conditional cases can cause the approach to adapt to a new situation. These cases are captured in the

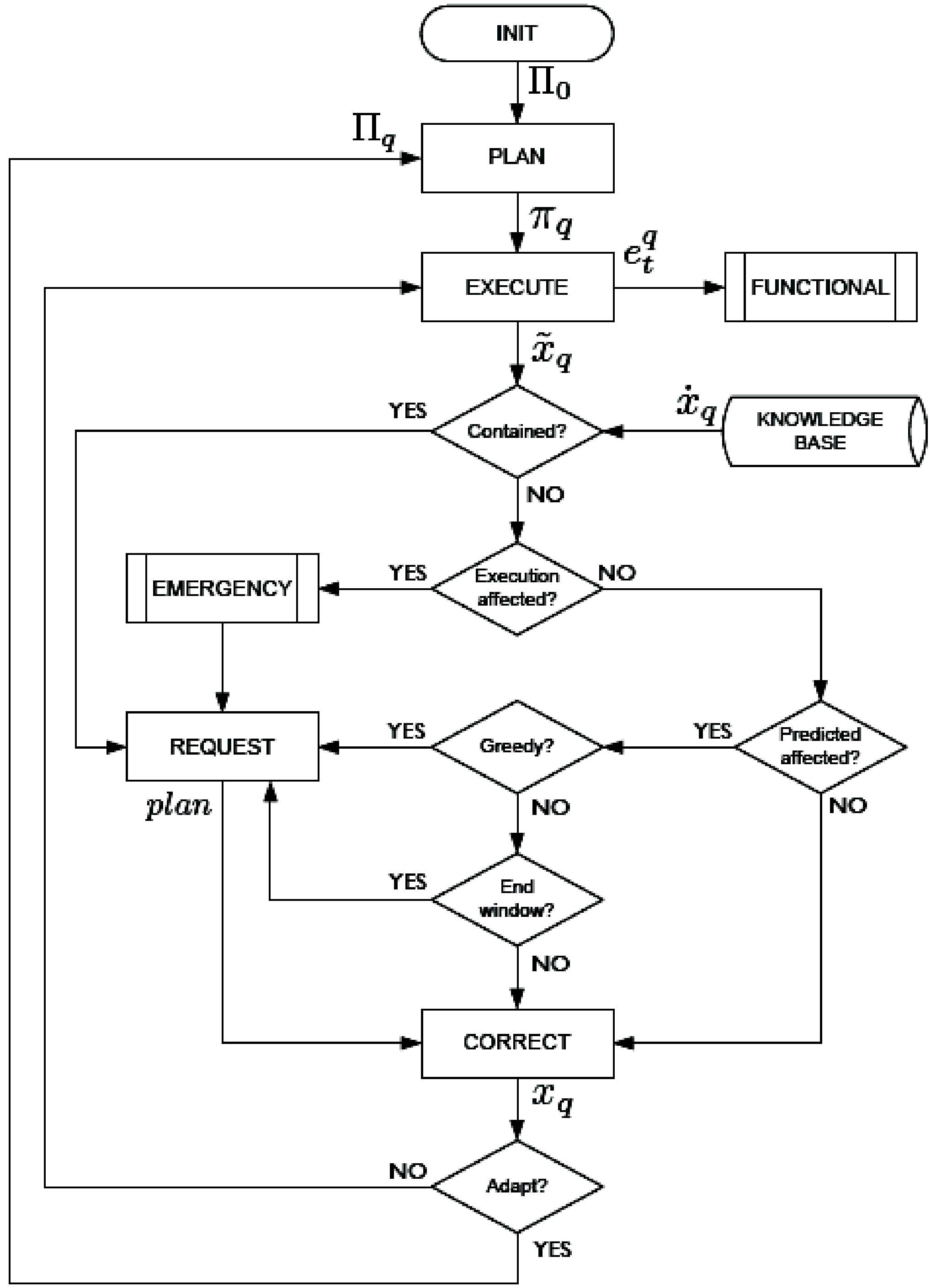

Figure 3: *Work flow of the adaptive mission planning approach: where* Π *is the mission environment,* π *the mission plan,* x *the status of the world and* e *the current action being executed.*

T	H	1	2	3	4	5
$\hat{D}_p$	0.18	1.00	0.41	0.15	0.20	0.20
$\hat{D}_s$	0.01	0.25	0.06	0.01	0.01	0.01
$PP_{0.5}$	0.90	0.37	0.76	0.91	0.89	0.89

Figure 4: *Normalised plan distance ($\hat{D}_p$), normalized state distance ($\hat{D}_s$) and Plan Proximity ($PP_{0.5}$) to a baseline plan π_0 that corresponds to the human driven mission for the static environment. Results in the partially-known dynamic environment of the human driven mission H and the different approach strategies obtained using $T \in [1;5]$, where T is the planning window horizon (number of atomic actions).*

figure by the conditional boxes: if the predicted state does not contain the state provided by the knowledge base, a new mission plan needs to be calculated. Additionally, if the execution is affected by the state provided by the knowledge base, an emergency behaviour is necessary. A new plan also needs to be calculated if actions calculated ahead in the current plan are affected by the state provided by the knowledge base or the end of the current plan is reached. Under all these circumstances, an adaptation process is triggered sending the loop back to the PLAN stage.

The combination of a finite approach with classical search provides computational efficiency: by keeping track of the requirements and expected effects of the actions planned ahead, we are able to forecast the impact of sensed events in the precomputed plan and, if necessary, react in advance. This reaction can be greedy or lazy. The greedy approach recalculates the plan as soon as the changes have been detected. Thus, it provides an optimal approach. Under a lazy behaviour, adaptation is delayed: planning is only performed at the end of the current window of execution. Thus, it provides a pseudo-optimal solution. Our implementation is able to handle temporal planning with durative actions, metric planning, opportunistic planning and dynamic planning.

Using the Plan Proximity metric presented in [41], the approach was evaluated in simulation under a static scenario and a partially known dynamic scenario, as outlined in [40] (see Fig. 4). The comparison results showed a high degree of similarity between our approach and the human driven adaptation. We have applied this approach to control the decision making process of the AUVs. Under this scenario, platforms are considered self-interested benevolent agents, able to react to the new knowledge provided by other vehicles when their capabilities match the needs of the mission. The collision of task allocations is unlikely and only possible with vehicles sharing similar capabilities. This approach optimizes the management of heterogeneous assets and resources by coupling resource capabilities and mission requirements in real time. It also provides a higher level interaction with the operator than current approaches. It is able to provide fast dynamic response to sensed events. It is suitable technique for long term deployment, where the initial goals of the mission are not fully explicit at the beginning and can change over time.

4.2 World Modeling

The most common storage method for a large corpus of knowledge on a computer system is a database, which is built from tables of values tied together with simple relations. The Multi-Dimensional Dynamic World Modelling study developed by SeeByte under the MoD Battlespace Access Unmanned Underwater Vehicles contract [42] utilised such a relational database. This showed the power of a loosely decoupled world model with multiple levels of data refinement within a single vehicle. Ontologies allow richer data structure than databases, with the description of hierarchies of classes that can posses both data values, and relations to other classes. Thus they are the obvious choice for providing structure in a complex world model.

The Semantic Web ontology language, OWL [43], has emerged as the recent de facto ontology language. With no compelling reasons to break from the status quo, OWL will be used in the ontologies for this project. OWL has three sub languages: OWL-Lite, OWL-DL and OWL-Full; each with increasing levels of expressiveness. OWL-Lite allows a classification hierarchy and simple constraints. OWL-DL is a super set of OWL-Lite, allowing more expressive ontologies, whilst still remaining decidable (all computations take a finite amount of time). OWL-Full is the most expressive dialect, but does not allow complete, decidable reasoning. Thus most applications of OWL utilise the OWL-DL sub language; we will follow suit, and any reference to OWL from this point on shall be taken to mean OWL-DL.

The two most popular ontological reasoning engines for OWL are RacerPro [44] and Pellet [45]. RacerPro is a closed source commercial product written in C++, with free academic licenses available. It is stated to provide support for reasoning over OWL ontologies. Pellet is open source software written in Java, and is able to reason over OWL ontologies.

Jena [46] is an open source Java framework that allows access to data stored in OWL ontologies. Queries are performed in SPARQL [47], which is effectively the equivalent of SQL for ontologies. Whilst it includes a built in rule-based inference engine, Jena also supports the application of external reasoners. Jena allows ontologies to be loaded into a database system (e.g. MySQL or PostgreSQL), for persistent storage and access. Information is stored as triples within the database.

The Pellet reasoner integrates well with Jena and the database storage back end. Unfortunately RacerPro does not have the same convenient interface to Jena, perhaps because it is written in C++ instead of Java. KAON2 [48] appears to be the only suitable alternative to

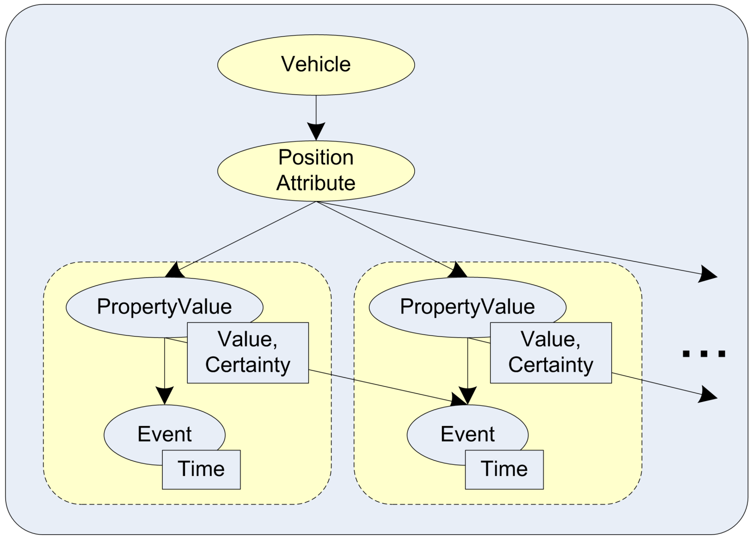

Figure 5: *Example of temporal data storage for vehicle position.*

Jena, also written in Java. It is free for academic use, but commercial use is subject to a paid license, as with Racer. KAON2 does not enjoy as much community support as Jena, but more importantly does not share the same level of integration with relational databases for persistent data storage.

For the above reasons, the tools chosen for this project are the Jena framework for interfacing with ontologies, and Pellet for reasoning.

4.2.1 Temporal Context

A good discussion of adding temporal context to ontologies is given in [28]. Their proposed solution is to create an instance of a PropertyValue class that contains the value (e.g. the sensor reading), and associate that with both the object attribute the value refers to (e.g. vehicle position), and start and end events for the value, which give time bounds to its validity. The intention is to use a similar scheme for this project. Figure 5 gives an example of vehicle position data stored in this way. The ovals represent ontology class instances, the square boxes are data values, and the arrows show instance relations. Each piece of temporal data will be stored as indicated in the dashed boxes. Relations to end events may be used if they are deemed to be advantageous.

4.2.2 Uncertainty

To cope with uncertainty, [28] also include a certainty parameter in their PropertyValue class. This could be a probabilistic measure or a variance, to be taken into account when using the value, such as in a sensor fusion process. The exact form the uncertainty is specified in will depend on the value in question, and will be decided when the ontologies and data processing algorithms are created. Figure 5 gives an example of vehicle position data stored with uncertainty, and temporal context.

4.2.3 Vehicle representation

For each vehicle in the system, the ontological representation will include details such as mass, water displacement volume, etc. It is assumed that these will not change over time, at least not during a mission. For vehicle attributes where temporal context is relevant, such as three dimensional position and pose, the representation discussed in section 4.2.1 above will be used. That is, attributes such as PositionAttribute and PoseAttribute will be associated with the vehicle, and PropertyValue instances created whenever a new value is needed.

Internal vehicle state will include concepts such as communications availability, power levels, and plans. Communications availability will record the state of links with the vehicle's peers at a given time; whether a link is up, and the estimated bandwidth available. Power levels will be a measure of the remaining energy reserves in the most appropriate form, be that time, watt-hours, or litres of fuel.

Representing the vehicle's plans is more involved, however. It also depends on what planning system is to be used, which is yet to be selected from the available systems under development in the Ocean Systems Laboratory. Thus we defer this decision until later.

4.2.4 Sensors and Sensor Data

Sensors will be defined as classes to be instantiated and associated with individual vehicle instances. Each sensor will include attributes such as relative location and orientation on the vehicle. As sensor instances are associated with vehicle instances, sensor data will also be associated with the instrument that generated it. This is explained diagrammatically in Fig. 6.

Large quantities of raw sensor data, such as sonar images, will be linked into the ontology as references to files on disk where the real data resides. Ontologies are not intended for storing such large quantities of data directly, and performance would be degraded otherwise.

4.2.5 Environment and Other Objects

Relatively small physical entities in the vehicle's environment can be modelled as simple point objects, e.g. a possible mine target, with a three dimensional position in space. Objects such as pipelines will require a slightly more complex representation. It is expected

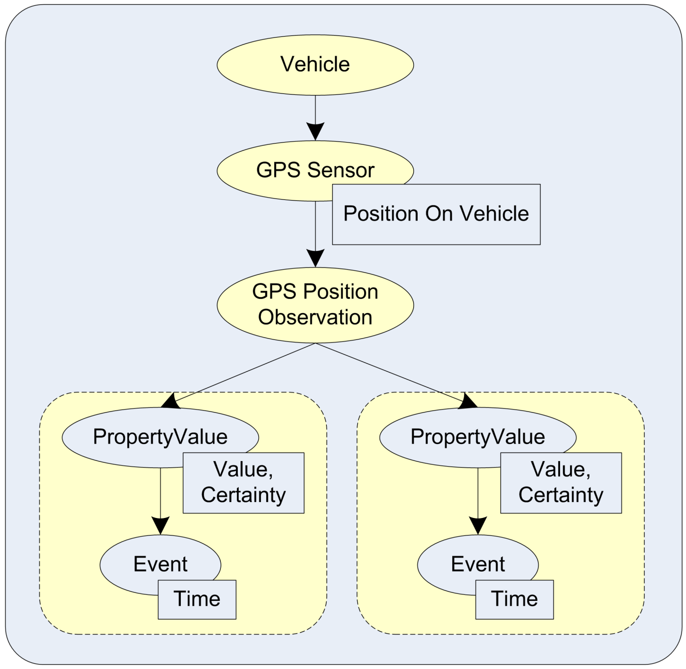

Figure 6: *Representation of vehicle GPS sensor and observations.*

that a collection of points will be used to represent approximately linear objects; a geometric line representation cannot be used as they are not necessarily straight. However, we must differentiate between multiple measurements of the same point, as for a point object, and multiple measurements of non-unique points, as in a linear object. To achieve this, different attribute classes for the measurements will be used. Multiple non-unique measurements will be associated with attributes of the form MultiPositionAttribute, whereas measurements for a unique point will simply use PositionAttribute.

3D surfaces of arbitrary objects may be represented using the same MultiPositionAttribute approach, which would provide a point cloud describing the object. This would allow entities such as the sea bed to be modelled. Of course issues of spiralling data storage requirements instantly spring to mind; this will be considered in the design of modules which access and maintain the data repository.

4.3 Software Architecture

A high level diagram of the software architecture is given in Fig. 7. The individual modules in this diagram are described in the sections below.

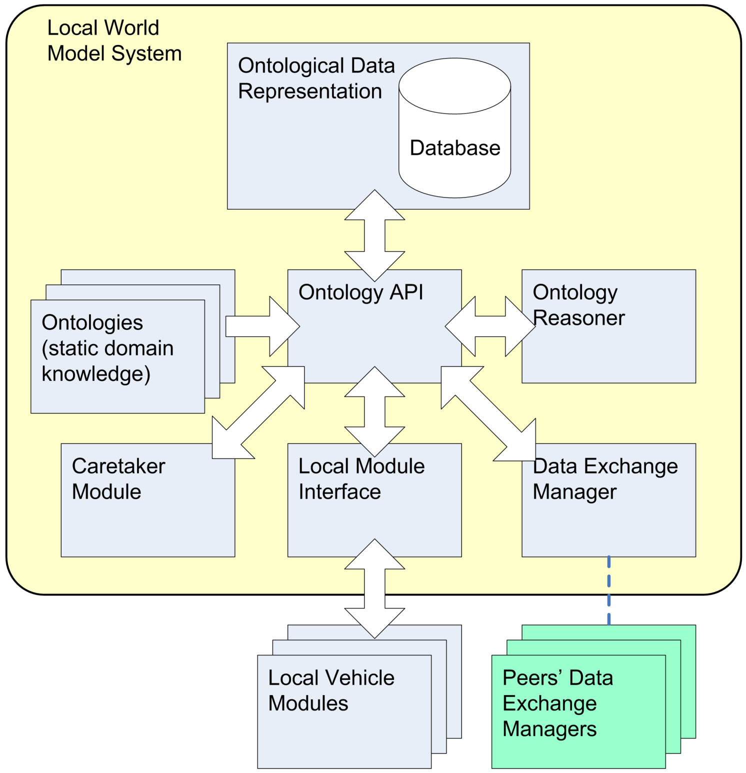

Figure 7: *Proposed world model architecture*

4.3.1 Ontology API

The Jena API [46] is a mature, well supported framework for working with ontologies, and will be used as a central point in the system. At the back end it will provide loading of static domain knowledge from OWL [43] formatted files, and the storage of dynamic information in the data repository using a standard database protocol. Support for consistency checking of the knowledge base and inference over the data will be provided by the Pellet [45] reasoner, interfacing through Jena. Finally Jena's client side API will be used by the caretaker module, local interface module and data exchange manager, discussed below.

4.3.2 Static Domain Knowledge

Knowledge about the vehicles and their environment will be input to the system as a set of ontology files in the OWL format. Before the ontology files are provided to the vehicle systems, a tool will be used to generate globally unique identifiers for each of the instances and relations described in these ontology files, and store these identifiers as labels attached to the concept they describe. These will provide a concise shared vocabulary with which to refer to concepts in the world.

4.3.3 Data Storage

It is intended that an SQL database back end will be used to store the ontological data created by the Jena API. Jena already includes facilities to interface with the open source MySQL and PostgreSQL databases; these will be evaluated to determine which is most appropriate.

4.3.4 Ontology Reasoner

The Pellet reasoning engine will allow consistency checking and inference to be performed on the local world model. The interface with Pellet is already implemented in the current Jena release.

4.3.5 Caretaker Module

To maintain the data storage within the hardware platform limitations, a caretaker module will be running in the background to archive or delete old data as appropriate. This will access the ontology API directly. The algorithms to be used by the caretaker module are yet to be defined. However, there will be some commonality with the mechanisms used by the data exchange manager, as both are concerned with the utility and timeliness of the information.

4.3.6 Local Module Interface

The Local Module Interface will serve as a gateway between sensor, processing and control modules on the vehicle, and the local world model. This interface code will also be responsible for ensuring world model consistency is maintained after writing data from the local modules, and that accesses by the multiple modules are correctly synchronised. As a performance enhancement, it may also be necessary for this module to cache data to improve query speed.

In order that existing vehicle modules do not need to understand the details of the ontology structure, small plug-able blocks of code will be used in the local module interface to convert between existing OceanSHELL [1] data messages and data structured for the ontology. For example, vehicle position messages could be simply converted to vehicle positions and property values for the ontology (see section 4.2.4). In the opposite direction, it is proposed that similar plug-able blocks will register interest in a type of information with the local module interface, and then when new data becomes available, appropriate OceanSHELL messages are sent out.

1. OceanSHELL is the lightweight UDP-based message protocol used in the Ocean Systems Laboratory for inter-module communication.

Active querying of information from the world model does not fit quite so well into the existing OceanSHELL push-based methodology. Initially, however, a function-call interface will be used for this, and a message based interface added later if it is found to be necessary.

4.3.7 Data Exchange Manager

A key part of the architecture is the Data Exchange Manager, which will orchestrate the exchange of information between the local world model for this vehicle, and those of the other vehicles in the collective. Tasks performed by the Data Exchange Manager include the selection of information to send to other vehicles based on utility and communications availability, and the merging of received data into the local world model whilst maintaining consistency. When accessing the local world model, the Data Exchange Manager will use the ontology API. Some of the issues surrounding the exchange of world model information between vehicles are discussed below.

- **Access control of broadcast medium**
  For communication between autonomous vehicles, wireless media are generally used, such as radio waves through the air, or acoustic signals underwater. These are broadcast media, so a mechanism for media access control (MAC) is required to reduce transmission collisions. For the purposes of this research it is assumed that a MAC facility is in place, such as in [49].
- **Bandwidth available per vehicle**
  In a simplistic broadcast scenario, if each vehicle is to be given an equal portion of bandwidth, on average the bandwidth available will be inversely proportional to the number of operating vehicles. Communications bandwidth is especially limited in the underwater domain that this project focuses on; typical throughput is of the order of a few kilobytes per second. Thus it is essential to make the most effective use of this bandwidth.
- **Consistency/integrity of the knowledge base**
  When information is received from other vehicles, care must be taken to ensure that the vehicle's local world model remains consistent, without dangling links, etc. A simple example would be where a block of information is to be exchanged, but it will not fit in a single message, and when split, each part is incomplete. Upon receipt of part of this information block, the data exchange manager must cache it until a whole consistent block has been received.
  The question is, who determines that the block is complete? The transmitting data exchange manager, which would convey the boundaries of the block to the recipient; or the receiving data exchange manager. It is proposed that a combined approach is taken. Firstly, the sender must delimit the boundaries of a complete block, to be understood by the recipient. This handles the issue of consistency within the block, by requiring the whole block to be present before insertion. Secondly, the recipient is responsible for checking that any dependencies with other instances in the distributed world model are satisfied before the block is inserted into the local world model. The later check ensures

that, for example, the sensor instance or even the vehicle that the information relates to is present in the local world model. Should this check fail, the local data exchange manager could either discard the block, or request the instance information it is lacking.

– **Selecting information to be shared**
The type or level of information to be exchanged will depend both on the needs of the recipients, and the capabilities of the communications channel. Given less communication bandwidth, either less frequent information or more concise, higher level information must be transmitted. Another factor is the stealth status of the vehicles. If performing a covert mission, very little or no explicit communication will be possible. It is intended that information on the needs, capabilities and status of links to all the peers be maintained by the data exchange manager, using the world model for storage where appropriate. The data exchange manager will utilise and update this information during its communications. The algorithm to choose which data to transmit and when is still to be determined.
To perform selective push-based distribution, the sender must have some concept of the value of the information to its peers. To this end, each platform maintains its own set of information needs, where each need specifies a named attribute of either an ontological class or specific class instance, referenced by UID. For example, one information need could be "position of object of interest", and another "classification of object of interest". Each information need has an associated priority, which is used in the exchange selection algorithm. To bootstrap the exchange process, the system assigns an implicit need on all peers to receive the local node's information needs. This allows nodes to automatically discover the needs of their peers, while supporting online modification of information needs as the mission progresses.
Observations that match an information need are referred to as "exchange candidates". A weighted queue is used to rank the exchange candidates. Each time the local node's slot transmission time is reached, the following actions are performed:

1. Check for modified information needs, grouping the same needs from different peers.
2. If a new peer is added to a need, mark previously completed candidates as potentially up for retransmission.
3. Check for new exchange candidates matching needs, and ACK timeouts of previously transmitted candidates.
4. Remove candidates that are now known by all peers that have the associated need.
5. Increment the weight of all exchange candidates by need priority.
6. Iterate through candidate list, sorted by decreasing weight, adding candidates to the exchange packet until no more will fit.
7. Mark chosen candidates as in progress and start ACK timer.

Whilst many works in the literature are concerned with reliable acoustic exchange mechanisms, they operate mainly at the packet level. By performing selection for (re)transmission on exchange candidates, our system will only retransmit unacknowledged observations that have not become obsolete, or are known to have been successfully delivered by another peer. It also provides the option to seamlessly include repeated observations from another peer together with the node's own observations.

A simple packet structure is employed: an acknowledgement block followed by an inform block containing the encoded exchange candidates. Information is encoded in the inform block using a tag-value structure, with tags for class instances, properties between instances, and primitive data-type properties. With the restriction that the information to be exchanged consists of a set of acyclic trees, the encoding process uses a stack to avoid the inclusion of the subject UID for every relation. Stack pop command tags are included where necessary. A little-endian binary encoding is used for the data type properties.

- **Reliability of data exchange**

  Wireless communication systems usually have some degree of loss; nowhere is this more true than of acoustic underwater communications. Data exchange between local world models must therefore be robust to this. Where information such as a regular status update is sent, it may be that data loss is merely tolerated. In the case of critical non-repeating transmissions, loss must be detected and remedied. The information exchange protocol will include the facility to request an acknowledgement of receipt, which will be used where necessary. When an ack is not requested, it will be the sender's responsibility to periodically repeat transmission of important information.

- **Merging of received information into local world model**

  It is not intended that information received from a peer vehicle will ever replace or overwrite information present in the local world model. Rather, each piece of information will be differentiated by its globally unique identifier (GUID), and merely add to the total knowledge store. For example, a possible mine is detected by a transit AUV and its presence recorded with an "unknown classification; the classification for this target could later be received from a hover-capable AUV, and added to the world model. This classification would be more recent, and thus take precedence. The origin of a piece of information may be tagged, to enable the flow of information to be traced. This would also prove useful in disregarding information if a particular vehicle is found to be malfunctioning.

- **Communications protocol and data encoding**

  In terms of communications protocol, the author considers agent communication languages such as KQML and FIPA-ACL to be unnecessarily complex for this application. In particular, it is not intended that communication be considered a plan-able act for the purposes of this system, as this would tie the distributed world model too much to the planning system of the vehicle. Thus a simple packet based protocol will be created that allows ontological instance data to be transferred from one vehicle to another with the minimum of overhead, both of processing and bandwidth. Care will be taken to make the protocol extensible, to allow for future expansion of the system capabilities. Initially, the following message types will be created:

  1. Inform

2. Acknowledge
3. Request

It is expected that in general, information will be pushed with inform messages rather than pulled by request messages. This is mainly intended to reduce communication bandwidth requirements in the very restricted underwater environment. It is easier and quicker for the originator of a piece of information to disseminate it using the broadcast medium than it is for all agents to repeatedly query each other for pertinent information. Of course the originator must have some concept of the utility of the information to its peers - one of the aims of this system. The request message will really serve as a backup to the pushing of inform messages, in the case that a vehicle is waiting on one key piece of information that it has not yet been sent.

By using a shared ontology base as a reference point, it should be possible to make the data exchange efficient whilst remaining generic. As mentioned in section 4.3.2, each class and relation in the shared ontology will be labelled with a GUID, so it will be sufficient to merely use these GUIDs as a shared vocabulary. Additionally, whenever class instances are created on a vehicle (such as for an altimeter reading), they will be tagged with a GUID, to ensure there will be no clashes between vehicles upon data exchange. Preliminary investigations have found a size of 80 bits to be suitable for these GUIDs. Given that the GUIDs and encoded values will generally contain significant redundancy, compression could be applied to reduce the size of the encoded data and fit more into a single acoustic packet. However, to produce the required packing effect, the compression implementation must be able to report the compressed size as each exchange candidate is added to the packet. The lack of this facility in commonly available compression libraries has prevented the implementation of this feature at present, but with sufficient programming effort this feature could be added. In whole-packet compression tests, packets were reduced to approximately one third of their original size, which would be a very welcome saving.

We now consider the structure of the individual message types.

Inform message

The inform message header will contain the following information:

– Originator ID
– Message ID
– Time stamp
– Ack required (boolean)

The actual information to be exchanged follows this header. The information in the world model will be comprised of instances of classes, relations between these instances, and primitive data type properties associated with class instances. It is proposed that these will be encoded as below.

Type: instance	Class GUID	Instance GUID	
Type: object property	Property GUID	Source GUID	Target GUID
Type: data type property	Property GUID	Source GUID	Data ...
Type: begin block	Block ID		
Type: continue block	Block ID	Segment No.	
Type: end block	Block ID	Segment No.	

In the case of the data type property, it will not be necessary to explicitly convey the type of the data that is to follow, as this is defined in the shared ontology, and referenced by the property GUID. Where the data is of variable length, such as a string, the length will be stored in the message before the actual data itself. The begin, continue and end block types are there to allow the sender to delimit a complete block of information with inter-dependencies, spanning multiple messages. They must be the first item in a message after the header, and their presence means that any information in that message is associated with the specified block.

Acknowledge message

This message will acknowledge the receipt of inform messages. The header will contain:

– Originator ID
– Time stamp

Then one or more blocks of the following form will be present:

Type: inform message ack	Message ID
Type: inform block ack	Block ID

This will allow multiple messages from different senders to be acknowledged in one message, potentially saving valuable bandwidth.

Request message

As stated above, there may be times when a vehicle will need to explicitly request information. The required content for this message is slightly less clear cut than the previous two. In order that the request not become incredibly detailed, it will be limited to specifying certain attributes, at a certain time (see section 4.2.1). A recipient of the request will then return the information it holds that best matches these criterion (in an inform message), or give no reply if it has no suitable information. The request message header will be the same as the inform header, but the message will contain multiple blocks of the form:

Type: request attribute	Attribute instance GUID	Time stamp

The attribute instance GUID will uniquely identify an attribute associated with a particular object instance (e.g. the GPS sensor on vehicle A). If the most recent information is required for an attribute, then the time stamp for that request block should be set to zero.

Further types of request blocks may be added to the message definition in future.

4.3.8 Data Processing

The world model will contain data at various levels of refinement/abstraction, from raw sensor data, though fused data and up to meta data about the vehicles and their status. As in the BAUUV project [42], an important component of the system will be modules that process information into other forms as needed, particularly the extraction of low bandwidth meta-data from raw sensor data.

This processing may be performed preemptively for some sensor data, such as detecting features in raw sonar data at acquisition time. Other queries, such as for an interpolated vehicle position, could be serviced as needed to prevent excessive growth in the size of the data store. In the current architecture, data processing resides outside the world model, with the other local modules that generate and consume data.

4.4 Information Representation

In this approach, all vehicles in a group share a set of ontologies defining the concepts in the domain, which serve as an extensible common vocabulary for exchange of semantically tagged information. We create a one-to-one mapping of the lengthy ontological uniform resource identifiers (URIs) to shorter binary unique identifiers (UIDs). The UIDs are composed as follows:

Platform ID	2 bytes
Time stamp in seconds	4 bytes
Count	2 bytes

Using a total of 8 bytes, this UID construction supports up to $2^{16} = 65536$ platforms, and the sequential count component allows generation of 65536 UIDs per second by each platform.

All observations in the world are stored as properties of a specific attribute, which are themselves stored as properties of a specific object in the world; this representation is derived from the situation awareness ontology described in [28]. Each observation includes a timestamp, to allow temporal sorting of information from different observers. An example ontology fragment for a pair of observations is given in Fig. 8.

4.5 Local Information Access

On each node of the network, the world model server runs as a separate process from other platform subsystems. A C++ client-side library provides multiple platform-local systems

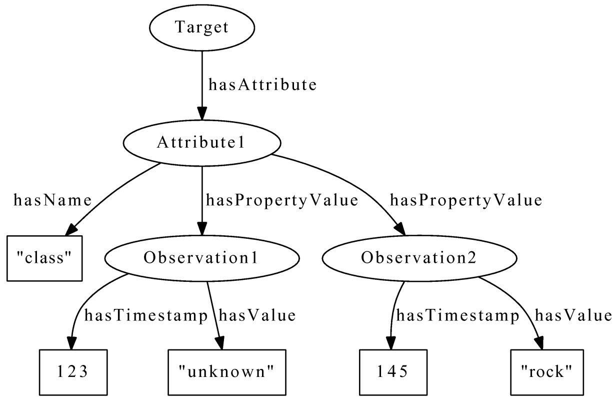

Figure 8: *Two example observations stored as properties of a single attribute.*

with access to the world model content. The facilities provided by this client library include:

– Retrieve object by UID
– Store one or more observations
– Create a network-efficient cache of select object attributes

The client library operates with user-supplied adaptor classes to map raw information trees to and from domain-specific C++ data structures. This provides access to domain-specific information in a structured way, with the benefit of compile-time checking on manipulation of these objects.

5 Conclusions

This report forms the first deliverable of the DRDC - Control Architecture for Multiple Autonomous Unmanned Vehicle Operations, under Contract #W7714-115079/001/SV, which constitutes the end of Phase I of the project.

The main goal of the project is to develop a multi-vehicle system, based on an adaptive mission planning system, interfaced with a World Model, based on Ontology representation. More in detail, during Phase II:

(a) We will develop and evaluate a mechanism for reliable, timely information distribution over high latency, low bandwidth, high loss, broadcast communication channels. The approach will use automatic dynamic routing to take account of unavailable channels due to range or noise.

(b) We will implement a distributed world model service within a service oriented architecture, that automatically updates local ontology descriptions of the environment and vehicles in an invisible and efficient way. The service approach is key to being efficient and timely.

(c) We will implement a mechanism to exchange information on a need to know basis, without explicit transmission scheduling by the mission executor/planner based on ontology labelling.

(d) We will implement a focus of attention mechanism that enables information exchange requirements to be modified on the fly during execution based on plan focus and events.

(e) We will demonstrate a user interface that can display hindcast, nowcast and forecast of vehicle behaviour and intentions from current world knowledge using models for prediction.

(f) We will derive performance metrics based on information exchange and on task performance to quantify the advances we have made.

(g) We will simulate a-e and demonstrate a-d above with an existing multi-vehicle mission planning and execution executive using *REMUS* and *NESSIE VI* autonomous underwater vehicles and WHOI acoustic modems at Heriot-Watt and in our Loch Earn trials site, Scotland.

References

[1] Miguelanez, Emilio, Patron, Pedro, Brown, Keith E., Petillot, Yvan R., and Lane, David M. (2011), Semantic Knowledge-Based Framework to Improve the Situation Awareness of Autonomous Underwater Vehicles, *IEEE Transactions on Knowledge and Data Engineering*, 23, 759–773.

[2] Guarino, N. (1998), Formal ontology in information systems, In *Proceedings of Formal Ontology in Information Systems*, IOS Press.

[3] Lesser, Victor R. and Corkill, Daniel D. (1981), Functionally Accurate, Cooperative Distributed Systems, *IEEE trans on systems, machines and cybernetics*, SMC-11, 81–96.

[4] Albus, James S. and Blidberg, D. Richard (1987), A control system architecture for multiple autonomous undersea vehicles (MAUV), In *Proceedings of the 5th International Symposium on Unmanned Untethered Submersible Technology*, pp. 444–466.

[5] Asama, Hajime, Matsumoto, Akihiro, and Ishida, Yoshiki (1989), Design of an Autonomous and Distributed Robot System: Actress, In *IEEE/RSJ International Workshop on Intelligent Robots and Systems '89. (IROS '89)*.

[6] Parker, Lynne E. (1994), Heterogeneous Multi-Robot Cooperation, Ph.D. thesis, Massachusetts Institute of Technology.

[7] Parker, Lynne E. (1998), ALLIANCE: An Architecture for Fault Tolerant Multirobot Cooperation, *IEEE Transactions on Robotics and Automation*, 14(2), 220–240.

[8] Jung, David, Cheng, Gordon, and Zelinsky, Alexander (1997), Experiments in Realising Cooperation between Autonomous Mobile Robots, In *The Fifth International Symposium on Experimental Robotics V*, pp. 609–620, Springer-Verlag.

[9] Burgard, Wolfram, Moors, Mark, Fox, Dieter, Simmons, Reid, and Thrun, Sebastian (2000), Collaborative Multi-Robot Exploration, In *Proceedings of IEEE International Conference on Robotics and Automation. (ICRA '00)*.

[10] Candea, Ciprian, Hu, Huosheng, Iocchi, Luca, Nardi, Daniele, and Piaggio, Maurizio (2001), Coordination in Multi-Agent RoboCup Teams, In *Robotics and Autonomous Systems*, Vol. 36, pp. 67–86.

[11] Bruce, James, Bowling, Michael, Browning, Brett, and Veloso, Manuela (2003), Multi-Robot Team Response to a Multi-Robot Opponent Team, In *IEEE International Conference on Robotics and Automation, 2003. (ICRA '03)*, Vol. 2, pp. 2281– 2286.

[12] Roth, Maayan, Vail, Douglas, and Veloso, Manuela (2003), A Real-time World Model for Multi-Robot Teams with High-Latency Communication, In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003. (IROS 2003).*, Vol. 3, pp. 2494–2499.

[13] Labrou, Jannis (2001), Standardizing Agent Communication, *Mutli-agents systems and applications*, pp. 74–97.

[14] Finin, Tim, McKay, Don, and Fritzson, Rich (1992), An Overview of KQML- A Knowledge Query and Manipulation Language, In J.M. Bradshaw, editor, Software Agents., AAAI Press.

[15] Genesereth, Michael R. and Fikes, Richard E. (1992), Knowledge Interchange Format Version 3.0 Reference Manual, Technical Report.

[16] Finin, Tim, Fritzson, Richard, McKay, Don, and McEntire, Robin (1997), KQML as an Agent Communication Language, Proceedings of 3rd International Conference on Information and Knowledge Management, pp. 456–463.

[17] FIPA (1997), FIPA 97 Specification Part 2 - Agent Communication Language Version 1.0, *http://www.fipa.org/repository/fipa97.html*.

[18] Walton, Christopher D. (2004), Multi-Agent Dialogue Protocols, In *Proceedings of the Eighth International Symposium on Artificial Intelligence and Mathematics, 2004*.

[19] Maio, Dario and Rizzi, Stefano (1993), Knowledge Architecture for Environment Representation in Autonomous Agents, In *Proceedings of the 8th International Symposium on Computer and Information Sciences*, pp. 4–11.

[20] Chandrasekaran, B., Josephson, John R., and Benjamins, V. Richard (1999), What Are Ontologies, And Why Do We Need Them?, *IEEE Intelligent Systems and Their Applications*, 14(1), 20–26.

[21] Kuipers, Benjamin (2000), The Spatial Semantic Hierarchy, *Artificial Intelligence*, 119, 191–233.

[22] Chella, Antonio, Cossentino, Massimo, Pirrone, Roberto, and Ruisi, Andrea (2002), Modelling Ontologies for Robotic Environments, Proceedings of the 14th International conference on software engineering and knowledge engineering, pp. 77–80.

[23] Hazarika, S. M. and Cohn, A. G (2002), Abducing Qualitative Spatio-Temporal histories from Partial Observations, In *Proceedings of the Eight Conference on Principles of Knowledge Representation and Reasoning (KR 2002)*, pp. 14–25.

[24] Kokar, Mieczyslaw M. and Wang, Jiao (2002), An example of using ontologies and symbolic information in automatic target recognition, In *Sensor Fusion: Architectures, Algorithms and Applications VI*, Vol. 4731, pp. 40–50.

[25] DARPA (2007), DARPA Agent Markup Language Web Site. http://www.daml.org.

[26] Schlenoff, Craig (2002), Linking Sensed Images to an Ontology of Obstacles to Aid in Autonomous Driving, In *Proceedings of the 18th National Conference on Artificial Intelligence: Workshop Ontologies for the Semantic Web*.

[27] Matheus, Christopher J. (2004), Using Ontology-based Rules for Situation Awareness and Information Fusion. Position Paper presented at the W3C Workshop

on Rule Languages for Interoperability, April 2005.
http://www.w3.org/2004/12/rules-ws/program2.

[28] Matheus, Christopher J., Kokar, Mieczyslaw M., and Baclawski, Kenneth (2003), A Core Ontology for Situation Awareness, In *Proceedings of the Sixth International Conference on Information Fusion*, pp. 545–552.

[29] Balakirsky, Stephen, Messina, Elena, Schlenoff, Craig, Smith, Scott, and Uschold, Michael (2004), Knowledge Representation for a Trash Collecting Robot: Results from the 2004 AAAI Spring Symposium, In *Robotics and Autonomous Systems*, Vol. 49, pp. 7–12.

[30] Ceruti, Marion G. (2004), Ontology for Level-One Sensor Fusion and Knowledge Discovery, 8th European Conference on Principles and Practice of Knowledge Discovery in Databases.

[31] Kim, J.H., Kim, Y.D., and Lee, K.H. (2004), The third generation of robotics: Ubiquitous robot, In *Proc of the 2nd Int Conf on Autonomous Robots and Agents*, Citeseer.

[32] Amigoni, Francesco and Neri, Mario Arrigoni (2005), An Application of Ontology Technologies to Robotic Agents, In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pp. 751–754.

[33] Mendoza, Rogan and Williams, Mary-Anne (2005), Ontology Based Object Categorisation for Robots, In *AOW ’05: Proceedings of the 2005 Australasian Ontology Workshop*, pp. 61–67.

[34] Schlenoff, Craig and Messina, Elena (2005), A Robot Ontology for Urban Search and Rescue, In *KRAS ’05: Proceedings of the 2005 ACM workshop on Research in knowledge representation for autonomous systems*, pp. 27–34.

[35] Schlenoff, Craig, Washington, Randy, and Barbera, Tony (2005), An Intelligent Ground Vehicle Ontology for Multi-Agent System Integration, In *Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pp. 169–174.

[36] Turner, R. (2005), Intelligent mission planning and control of autonomous underwater vehicles, In *Workshop on Planning under uncertainty for autonomous systems, 15th International Conference on Automated Planning and Scheduling (ICAPS05)*.

[37] Bellingham, J., Kirkwood, B., and Rajan, K. (2006), Tutorial on issues in underwater robotic applications, In *16th International Conference on Automated Planning and Scheduling*.

[38] Patrón, Pedro and Petillot, Yvan R. (2008), The underwater environment: A challenge for planning, In *Proceedings of the Conference of the UK Planning Special Interest Group (PlanSIG’08)*, Edinburgh, UK.

[39] Fox, M., Long, D., Py, F., Rajan, K., and Ryan, J. (September 2007), In situ analysis for intelligent control, In *Proceedings of the IEEE International Conference Oceans (Oceans07)*.

[40] Patrón, Pedro, Lane, David M., and Petillot, Yvan R. (2009), Continuous mission plan adaptation for autonomous vehicles: balancing effort and reward, In *4th Workshop on Planning and Plan Execution for Real-World Systems, 19th International Conference on Automated Planning and Scheduling (ICAPS'09)*, pp. 50–57, Thessaloniki, Greece.

[41] Patrón, Pedro and Birch, Alexandra (2009), Plan proximity: an enhanced metric for plan stability, In *Workshop on Verification and Validation of Planning and Scheduling Systems, 19th International Conference on Automated Planning and Scheduling (ICAPS'09)*, pp. 74–75, Thessaloniki, Greece.

[42] SeeByte (2005), BAUVV-DMDWM-SFR-01 - Dynamic Multi-dimensional World Modelling Final Report, Technical Report SeeByte Ltd.

[43] (2007), OWL Web Ontology Language. http://www.w3.org/TR/owl-features.

[44] (2007), RacerPro. http://www.racer-systems.com.

[45] (2007), Pellet: An OWL DL Reasoner. http://pellet.owldl.com.

[46] (2007), Jena - A Semantic Web Framework for Java. http://jena.sourceforge.net.

[47] (2007), SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query.

[48] (2007), KAON2 - Ontology Management for the Semantic Web. http://kaon2.semanticweb.org.

[49] Peleato, Borja and Stojanovic, Milica (2006), A MAC Protocol for Ad-Hoc Underwater Acoustic Sensor Networks, In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pp. 113–115.

This page intentionally left blank.

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.) Ocean Systems Lab Department of Electrical, Electronic and Computer Engineering School of Engineering and Physical Sciences Earl Mountbatten Building Heriot-Watt University Riccarton, Edinburgh, EH14 4AS Scotland	2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED (NON-CONTROLLED GOODS) DMC A REVIEW: GCEC JUNE 2010

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

Control Architecture for Multiple Autonomous Unmanned Vehicle Operations

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

Maurelli, F.; Cartwright, J.; Lane, D.M.

5. DATE OF PUBLICATION (Month and year of publication of document.) May 2012	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.) 42	6b. NO. OF REFS (Total cited in document.) 49

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.)

Contract Report

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

Defence R&D Canada – CORA
Dept. of National Defence, MGen G.R. Pearkes Bldg., 101 Colonel By Drive, Ottawa, Ontario, Canada K1A 0K2

9a. PROJECT NO. (The applicable research and development project number under which the document was written. Please specify whether project or grant.) 10bz04	9b. GRANT OR CONTRACT NO. (If appropriate, the applicable number under which the document was written.) W7714-115079/001/SV
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC CORA CR 2012-100	10b. OTHER DOCUMENT NO(s). (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.)

Unlimited

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.)

This report forms the first deliverable of the DRDC - Control Architecture for Multiple Autonomous Unmanned Vehicle Operations, under Contract #W7714-115079/001/SV. This project addresses the topic of a decentralised world model service that operates across multiple underwater vehicles. After an outline of the related literature, the report focuses on the open challenges and on the proposed approach. The main features are an adaptive mission planning system, interfaced with a World Model, based on Ontology representation. An important focus is given to the communication among vehicles, considering a temporal context and the restrictions of the underwater domain.

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 is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

auv
mission planning
world modelling
ontology
acoustic communication

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