Neptune

Technical Whitepaper

seebyte

1 INTRODUCTION

Autonomous underwater vehicles (AUV), and more recently unmanned surface vehicles (USV), have grown in acceptance to the point where their relevance for MCM applications and operations in littoral waters is unquestioned. Military uptake continues to increase; the number of navies operating Unmanned Maritime Systems (UMS) is now internationally widespread with dedicated teams trained to operate unmanned autonomous assets in combat situations. The UMS are also starting to prove capability in other applications areas such as anti-submarine warfare and intelligence gathering.

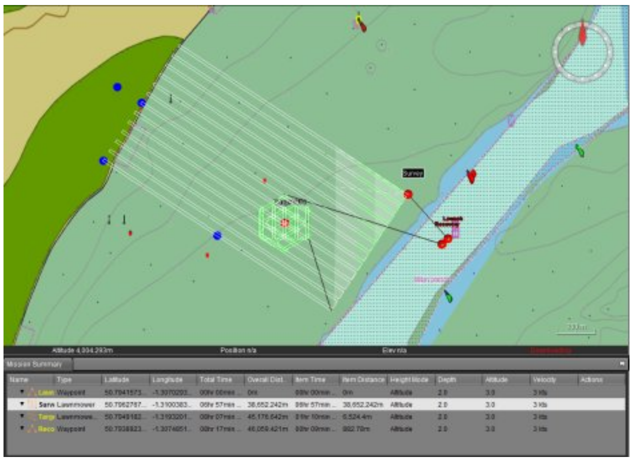
However, there are common limitations that often impede operations and prevent operators from using unmanned vehicles to their full potential. For example, a region survey to gather data should be capable of using all the vehicles regardless of their type. However, these vehicles have dramatically different characteristics (speed, endurance, depth rating, communication, payload type, navigation suite) and this presents significant challenges in the design of the system. Likewise, heterogeneous fleets of UMS require operators to be familiar with the particulars of each manufacturer's control system. As fleets expand, this can be costly and impractical.

As UMS become more widespread the market enjoys a positive relationship with advancements in technology designed to expand the capabilities of unmanned autonomous systems. Tools have been designed to enhance the autonomous capabilities of UMS, allowing vehicles to become increasingly operator independent. Additionally, software solutions have been designed to overcome some common limitations of UMSs; shared interfaces for multi-vehicle fleets allow for larger fleets and greater output whilst decreasing operator workload. This paper will outline the lessons learnt developing software that enhances the autonomous capabilities of UMSs, and discuss examples of where these software frameworks have been successfully demonstrated.

2 FROM WAYPOINTS TO AUTONOMY

There has been a paradigm shift from the early days where UMS missions were a set of simple pre-planned waypoints to today where specialist software houses and manufacturers are developing new concepts to make UMSs truly autonomous. There are three main threads to these developments which will be presented chronologically in this paper: true adaptive autonomy, where the vehicle is able to adjust its behaviour in response to feedback from the environment; collaborative or 'SWARM' technology which enables unmanned systems to work as part of an adaptive fleet with each asset communicating continuously with the others in the fleet to achieve a common goal; and over-the-horizon operations where the fleet is able to perform tasks autonomously supported only by a shore-side team to carry out launch and recovery.

The term autonomy is often applied as a sweeping term across a broad range of technology and covers a wide range of levels of autonomy. A popular view on autonomy is described in the ALFUS model where autonomy is defined by an UMSs own ability of sensing, perceiving, analyzing, communicating, planning, decision-making, and acting/executing, to achieve its goals as assigned by its human operator(s) through designed Human Robot Interface. The levels of autonomy are defined by the level of human/robot interaction that can vary due to mission and environmental complexity.



Name	Type	Latitude	Longitude	Total Time	Overall Dist.	Item Time	Item Distance	Height Mode	Depth	Altitude	Velocity	Actions
Laun	Waypoint	50.7941573...	-1.3070203...	00hr 00min ...	0m	00hr 00min ...	0m	Altitude	2.0	3.0	3 kts	
Surv	Lawnmower	50.7962787...	-1.3100383...	06hr 57min ...	38,652.242m	06hr 57min ...	38,652.242m	Altitude	2.0	3.0	3 kts	
Targ	Lawnmowe...	50.7949182...	-1.3193201...	08hr 07min ...	45,176.642m	01hr 10min ...	6,524.4m	Altitude	2.0	3.0	3 kts	
Reco	Waypoint	50.7938823...	-1.3074651...	08hr 17min ...	46,059.421m	00hr 09min ...	882.78m	Altitude	2.0	3.0	3 kts	

Before Neptune complex waypoint planning was the standard approach, presenting numerous opportunities for error

To start with, this advanced software processing can be as straight-forward as improving the visualization of the incoming data. The next step up in complexity, traditionally referred to as decision support, relies on the smart software to reduce the amount / complexity of data that the human operator has to view. This might mean writing an algorithm that can recognize and flag to the operator distinctive shapes and patterns so that the operator is able to make key decisions based on the relevant data.

Moving up again in sophistication, the software processing can be used to achieve "operator-in-the-loop" autonomy. This might be adaptive control of the vehicle while the operator monitors progress and intervenes where necessary.

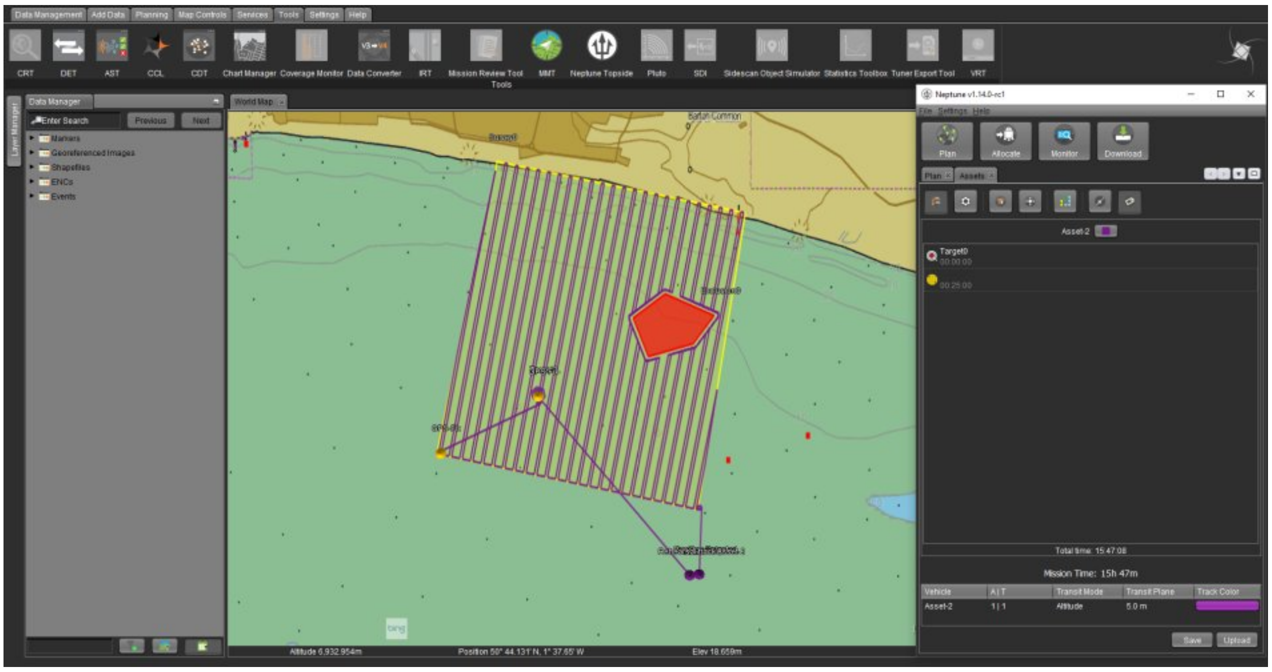
The final stage of smart software processing is for full autonomy without the operator directly involved. In robotics true autonomy occurs when a system is not only able to perform a series of actions un-aided, but is able to determine itself which action is appropriate according to the feedback it receives from its sensors. The main advantage to full autonomy is in situations where the vehicle has to operate for extended periods in a remote, hard-to-access or dangerous location.

From the perspective of end users, the ability to scale this capability from a single UMS to a fleet of assets is where the benefits become apparent. Managing a large fleet of UMSs typically requires a large support team responsible for the mission-planning, launch, recovery and exporting the data from the vehicle. Each vehicle will typically have a user interface unique to the manufacturer. This means that a mixed fleet of UMSs cannot be operated from a common interface. With heterogeneous fleets of UMSs, operators are required to be familiar with the particulars of each manufacturer's operating system. This is unsustainable as the fleet grows in numbers. These limitations can often result in less than satisfactory area coverage and data collection.

To create an autonomous ‘swarm’ or fleet of vehicles, it is essential that they are able to communicate with each other through a shared ‘language’, regardless of vehicle make or model. This can be achieved by integrating the vehicles into the Neptune autonomy architecture. Underpinned by a modular architecture, this model provides a simple way to organize the software modules, interfaces and communication protocols in an autonomous system. The architecture is specifically designed to allow the adaption, upgrade or replacement of components. This effectively enables fleets of heterogeneous assets to be managed and monitored from a single interface. By providing a shared autonomy architecture for fleets of assets, each asset is able to present its capabilities to the mission planner and undertake relevant behaviours accordingly.

Neptune's decentralised autonomy architecture allows multiple tasks to be run in parallel with the available vehicles automatically taking responsibility for tasks. This acts as a force multiplier as each vehicle is able to benefit from all the information of every vehicle in the fleet meaning that they are able to cover far greater areas at a time and the operation is not affected if a vehicle malfunctions.

This give rise to the third and final thread of autonomy: over-the-horizon operations. The crux of this approach relies on goal-based mission planning; the fleet is assigned a task or tasks to accomplish by the operator pre-deployment but software decides the optimal approach based on the feedback from the vehicle payloads. The technologies developed provide the first major steps towards a paradigm shift: a move away from men on the front-line operations towards unmanned over-the-horizon multi-squad operations supported by a shore-side team.



Vehicle	A | T	Transit Mode	Transit Plane	Track Color
Asset-2	1 | 1	Altitude	5.0 m	🟪

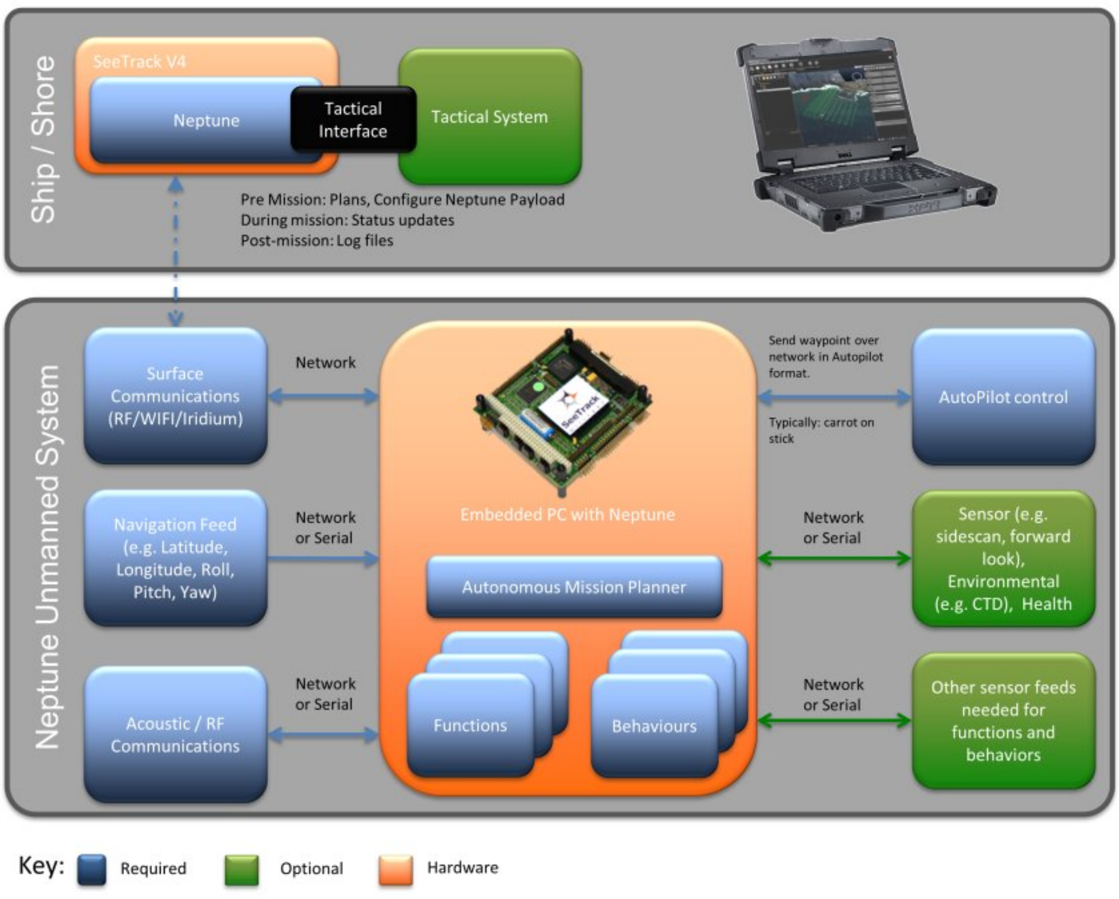
Neptune Goal Based Planning, where the operator only has to enter the goals, such as the regions to be surveyed and the exclusions zones to be avoided

3 ARCHITECTURE DESIGN

The SeeByte Neptune Autonomy Architecture has both on-board (embedded) and top-side (external interface) components, allowing it to integrate this sequence of events with the system operator. Neptune utilises underlying payload control architecture (PCA) that benefits from modern layered data model design and middleware messaging technology. This PCA effectively provides a level of abstraction between the system hardware and control software, sensors and planning processes. Neptune then builds an autonomy engine that provides a modular software architecture capable of integrating third-party modules.

3.1 Top-side / User Interface

The top-side component shows the operator interface. This interface is for user entry of mission requirements and, where available, any prior knowledge of the world (World Model). After the user entry has occurred, the top-side then focuses on the production of the initial Mission Plan. This initial Mission Plan is provided to all vehicles which are being used in the mission. At mission start up, all vehicles are provided the complete plan and so are aware of the responsibilities of all other vehicles. The mission planner is suitable for single or multiple vehicle operations. The simplest approach for the planner is to provide a sequential list of tasks for each vehicle. However, the planner is capable of providing a list of tasks that the vehicle(s) can prioritize in an autonomous manner, providing far greater scope for in-mission autonomy. This ensures that the process is human understandable and is less reliant on communications during the mission.


Ship / Shore
SeeTrack V4
Neptune
Tactical Interface
Tactical System
Pre Mission: Plans, Configure Neptune Payload
During mission: Status updates
Post-mission: Log files
Neptune Unmanned System
Surface Communications (RF/WIFI/Iridium)
Network
Navigation Feed (e.g. Latitude, Longitude, Roll, Pitch, Yaw)
Network or Serial
Acoustic / RF Communications
Network or Serial
Embedded PC with Neptune
Autonomous Mission Planner
Functions
Behaviours
Send waypoint over network in Autopilot format.
Typically: carrot on stick
AutoPilot control
Network or Serial
Sensor (e.g. sidescan, forward look), Environmental (e.g. CTD), Health
Network or Serial
Other sensor feeds needed for functions and behaviors
Key:
Required
Optional
Hardware

Neptune System Architecture

The initial mission planning can use a Discovery service. The Discovery service conducts a mapping between the goals of the mission and the capabilities of the different vehicles. The system is able to quickly determine if the mission is possible and will then transmit this information to the operator. It is possible for each vehicle to be equipped with different sensors, different Functions - software modules such as Automatic Target Recognition (ATR) that process data - and different Behaviours - software modules that can take control of the vehicle. Each Function and Behaviour is defined by its Requirements and its Capabilities. For example, an ATR Function requires access to Sensor Imagery and Vehicle Navigation information; its capability is to detect and localize objects, a requirement for the autonomous Close-Inspection Behaviour for object Re-Acquisition to function. When multiple Functions are available to achieve a mission goal (for example, two different ATR modules are on-board the vehicle), the operator may select that one or all of the options are used. When multiple Behaviours are available for a mission task (for example different Close Inspection patterns) the operator must select which behaviour is used. This ensures only one autonomous process is controlling the vehicle at a time.

3.2 On-board / Embedded

The second component of the Autonomy Architecture is the on-board modules. These are responsible for the actual execution of the mission plan. The Mission Executive module is responsible for starting, stopping and monitoring each Behaviour's process on-board the vehicle. It uses the Mission Plan and World Model as inputs and calls the relevant Behaviours needed to execute its mission goals. It is the high-level controller, but the actual vehicle autonomy is provided by the appropriate Behaviours. As the complexity of on-board vehicle autonomy increases the Mission Executive takes responsibility for high-level tasks such as in-mission re-planning and dynamic goal reselection.

On-board the vehicles, the primary autonomy capabilities are carried out by the Behaviour and Function modules. Functions process the on-board sensor data and include capabilities such as ATR on sonar imagery and estimating sea currents. Behaviours typically take information from the Functions and use these to control the vehicle. An example would be conducting a Re-acquire mission relative to an ATR call or directing a Survey Region mission based on the estimated direction of the sea currents. The Behaviours are responsible for using the necessary Functions to obtain the information required to make a decision.

The mission plan created during the planning phase may be dynamically updated in-mission. Within the MCM context, a vehicle conducting a Survey Region mission that detects a mine-like target using a side-scan ATR Function can create a new objective, requiring that the target is Re-acquired. This would be added to the Mission Plan and would need to be transmitted to the other vehicles via acoustic communications. At this point, one of the vehicles with the appropriate capabilities would take responsibility for carrying out this goal.

3.3 Interfaces and Common Data Formats

The on-board architecture is open at three key points.

The highest level is the Mission Interface that allows third-parties to integrate with the core mission planning system. This allows progress reports on the current Mission Plan to be communicated and for adaptations or updates of the mission objectives and mission constraints to be passed down. This could be used, for example, to allow advanced in-mission dynamic planning to be utilized, if the operational scenario allowed this.

The intermediate Services Interface allows third-parties to replace or add new Behaviours or Functions. Behaviours and Functions use a common interface requirement so that the method for broadcasting their capabilities and requirements is consistent. These services are also required to broadcast status updates so that other software modules are aware of whether they are on, off, idle or processing. They are also required to broadcast their health status, allowing the other modules that depend on them to understand if they are performing correctly. This common functionality, ensuring inter-operability, is required for all new modules integrated into the system.

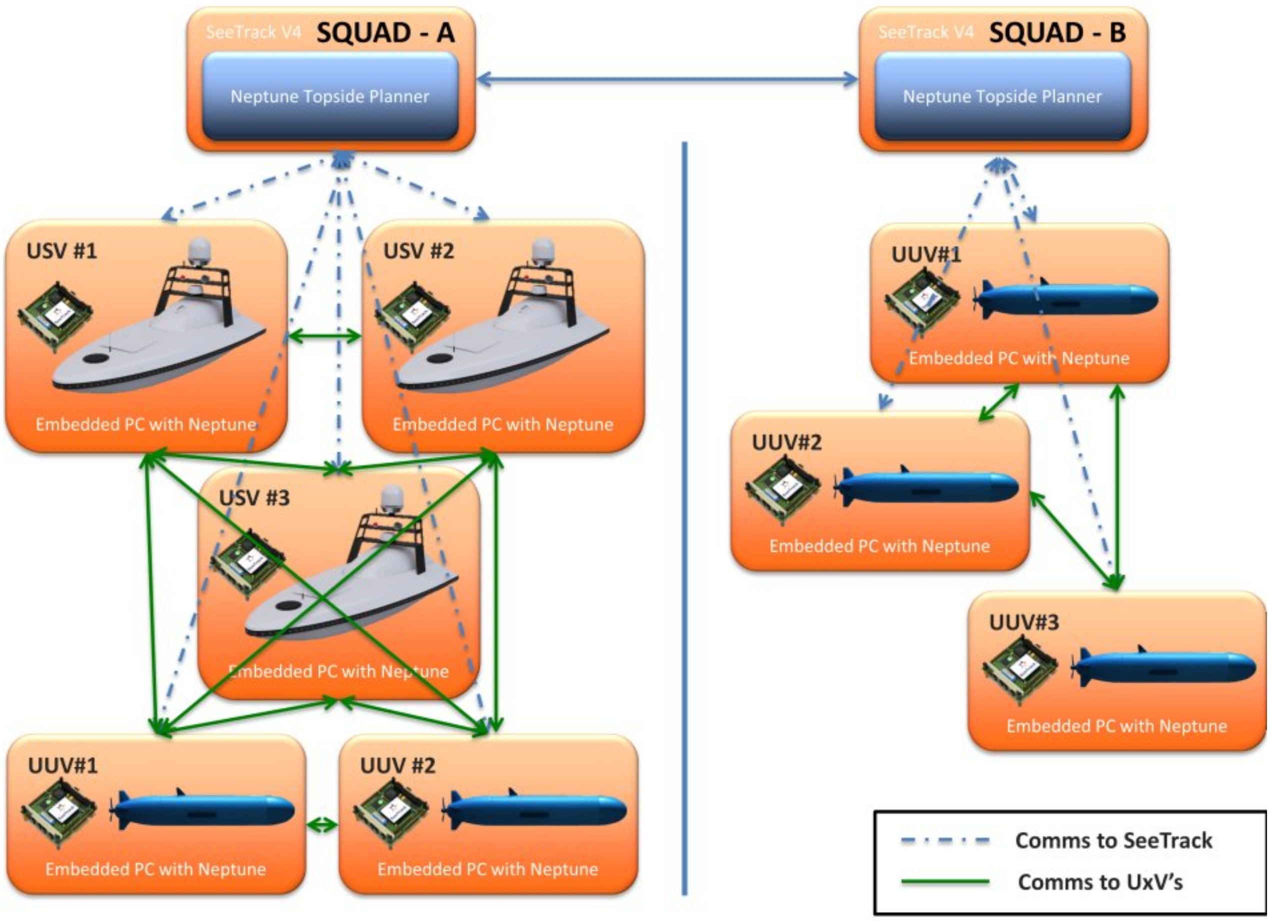
The lowest level is the Vehicle Interface. This allows new hardware components, such as vehicles, sensors or navigation systems, to be integrated. Integrating a known type of sensor, for example a side-scan sonar, is simple and requires that a specific driver module is developed that reads the sensor data into the appropriate pre-defined common data format. The common data format is then recognized by the autonomy system and can be used by all the higher level Functions and Behaviours. If a new type of sensor needs to be integrated, then a new common data format needs to be defined to allow the easy data interchange between all relevant Functions and Behaviours.

4 NEPTUNE OPERATION

The Neptune User Interface allows the user to specify Regions for operations to take place. Lists of mission objectives that form the Mission Plan are automatically built up. Once the plan has been produced, a period of Capability Discovery is executed to automatically determine the assets and sensors available. The list of mission objectives is then mapped to the available capabilities of the different assets. This allows a first phase of mission planning to take place pre-deployment, inlcuding a full mission rehearsal. This initial mission plan is shared with all vehicles across the squad that contain the embedded Neptune system (Figure 4). Multiple squads can be operated, and communication relays between squads and the command and control station hosting SeeTrack are supported. As this is a decentralised autonomy system all communication links are not assumed to be operational throughout the mission. An exemplar of this in operation can be seen in section 5.

If the Capability Discovery encounters multiple options, in either Behaviours or Functions, then the operator will be presented with a list of choices. In the case of a Behaviour, only one option may be selected – for example, the particular re-acquire pattern employed. In the case of a Function then it may be possible to select multiple options – for example, to run multiple object detection systems on the same data.

Once vehicles are deployed and undertaking the Mission Plan, each vehicle transmits a progress update on its current mission objective using the communications capability on the host platform (e.g. Iridium, Acoustic Communications). This information may be used to define no-go areas for other vehicles, so that



Neptune can used for multi-vehicle, multi-domain, and multi-squad operations

only one vehicle in the same domain carries out one objective within a region. For underwater platforms, to further minimize the chances of vehicle collision, vehicles move to different areas at allocated depths. The entire process is designed to work in the absence or sporadic availability of communications.

Dynamic mission planning may take place based on environmental parameters to ensure a vehicle carries out its current mission objective in the most efficient manner. For example, a vehicle surveying a region may modify the survey orientation based on sea current or wave direction information.

Another example of a dynamic behaviour would be if an object is detected by the on-board ATR while surveying a region. The Re-acquire of this target becomes a new, dynamic objective that one of the vehicles will execute, where only vehicles with the relevant capabilities will be allowed to take on this objective.

Where maritime operations can be broken into phases, Neptune provides built-in capabilities to address survey and re-acquire phases. New phases can be defined to extend the autonomy capability to cope with further operating scenarios such as the Neutralization phase in the case of MCM operations.

4.1 Behaviours and Functions

A Behaviour is defined as something that controls the vehicle. An example would be the Behaviour to execute a lawnmower pattern over a defined Region of seafloor.

A Function processes data on-board the vehicle. An example would be the Function to perform ATR on the incoming side-scan sonar data.

All Behaviours and Functions operate using the template Services Interface. Therefore, all of them must specify their capabilities and also set out their requirements in a common manner. This is to ensure that the system can autonomously determine if a mission is possible based on the available resources and current health status of the capabilities. For example, a vehicle health return to base behaviour will not be possible if a battery monitoring function is not available. This template approach provides the system with a clear method for describing capabilities and dependencies. It also provides a clear method for incorporating new Behaviours and Functions.

4.2 Existing Behaviours and Functions

The current Behaviours include simple actions, such as “Go to Waypoint”, as well as different survey behaviours, such as Lawnmower and Star pattern. The Behaviour must specify the input parameters it requires to run and its requirements from other Behaviours or Functions. This allows the system to determine if the mission objectives may be accomplished with the capabilities made available.

The current Functions include sidescan sonar ATR, system monitor, environmental monitor and communications manager. All Functions within the system must specify the input parameters and output parameters produced. All Functions and Behaviours also use an XML-based Configuration file system allowing them to be optimized pre-mission via the Neptune user interface.

4.3 Extensions

The Neptune autonomy architecture is designed to allow third-party integration at a variety of levels. Neptune has been specifically designed to allow very straightforward integration of third-party behaviours and functions. This is to allow rapid testing and deployment of key technology components without having to undertake any technical development on the core Neptune system.

The design of Neptune uses the standard three open interfaces, through which third-parties can add new capability:

- Mission Interface – this is to allow changes to the current Mission Plan from the SeeTrack.
- Services Interface – this is to allow new Behaviours and Functions, such as ATR, to be integrated. They will run embedded on the vehicle.
- Vehicle interface – this is to allow new hardware, such as vehicles, sensor data or navigation system, to be integrated.

In Neptune, data is stored and passed round the autonomy framework using a set of messages. New messages can be defined as required, for example, to hold forward look sonar or laser range finding. This extensibility is necessary to ensure that Functions using the incoming data stream are able to correctly interpret the raw data.

All components use an XML based schema for their configuration files. This allows the configuration for all components to be human readable by the user through the interface. This ensures that the user may check and, if necessary, modify parameters for the different Behaviours and Functions. This checking phase is optional but allows the expert operator to configure the specific settings of the different components.

4.4 Integrating new platforms

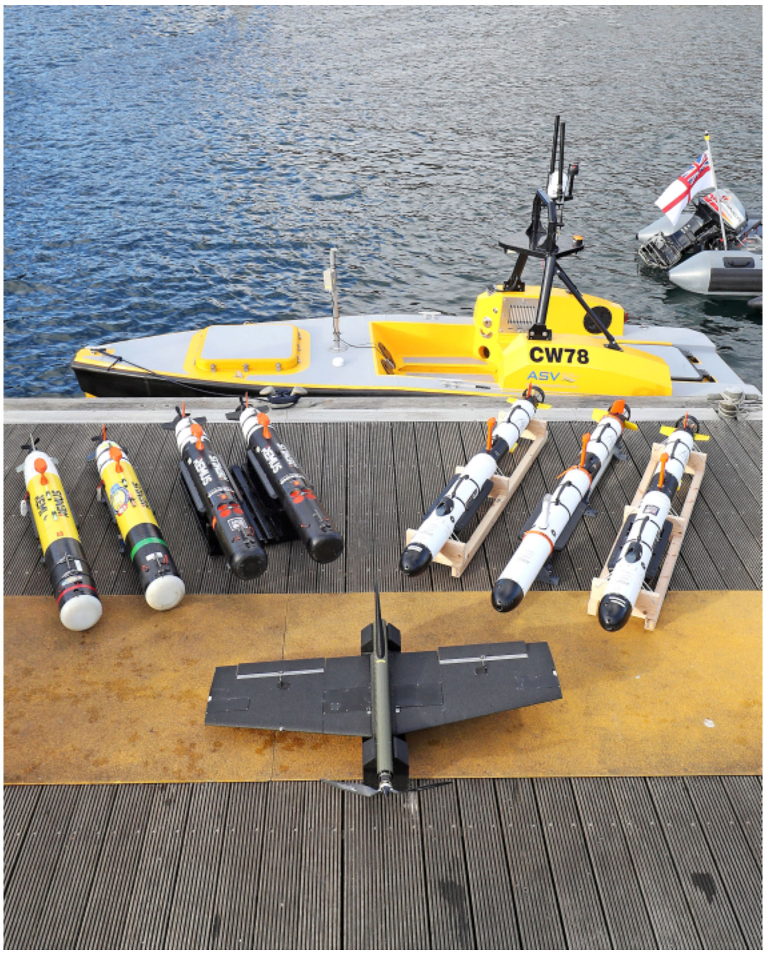
In order to integrate onto a new vehicle platform, a computer is needed to host the Neptune software. This is typically a small form factor computer such as an x86 based PC-104 or ARM based NVIDIA Jetson. This computer must be able to run a flavour of the Linux Operating system (typically Ubuntu). The minimum interfacing requirement that must be done with the vehicle is as follows:

- A communications link for pre-mission planning with the Neptune top-side
- Navigation feed (Latitude, Longitude, Roll, Pitch, Yaw)
- Ability to send waypoints to the vehicle’s autopilot system
- Ability to configure and start data capturing of sensors

5 AUTONOMY IN ACTION

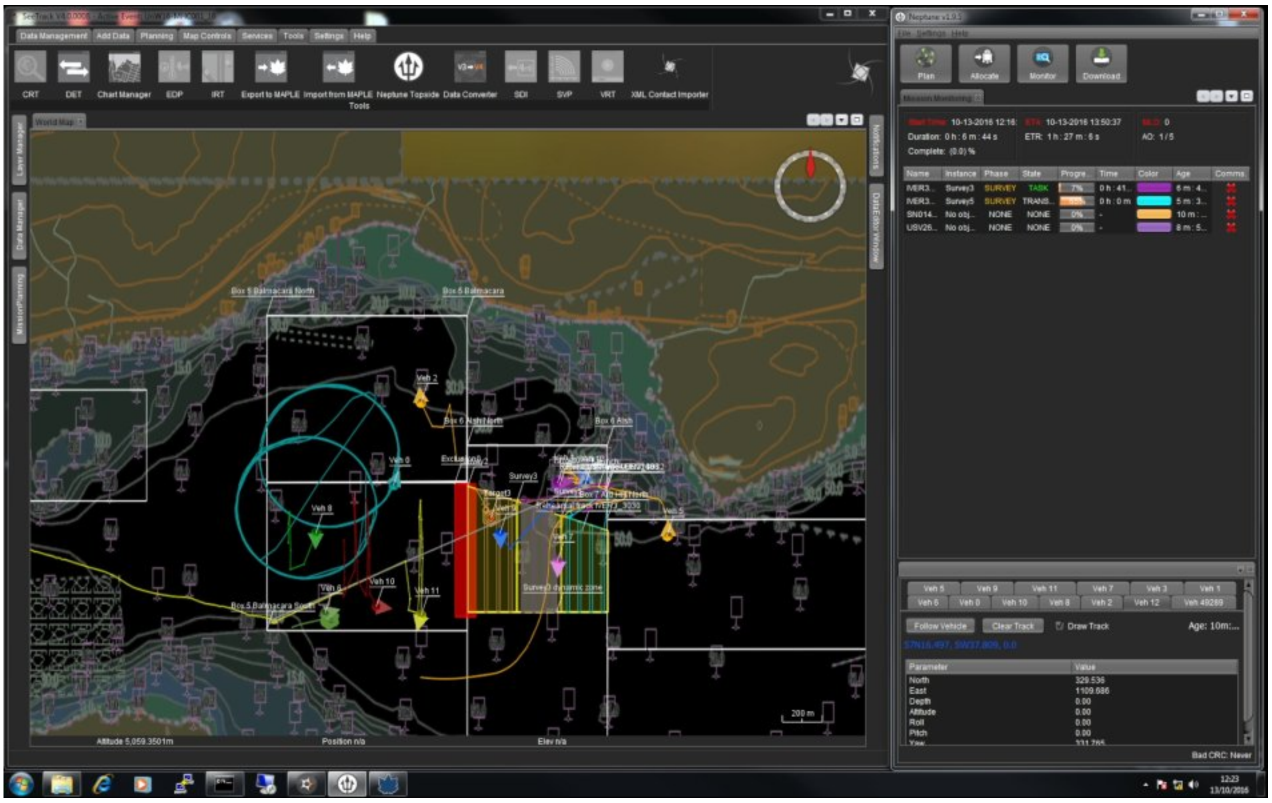
Building upon the developments in adaptive autonomy, these can be scaled up to give rise to truly adaptive autonomous fleets of UMS. Essentially, operating as part of a fleet means that a goal or mission, which would potentially be beyond the capabilities of a singular UMS and its sensors alone, can become possible.

SeeByte partnered with primes including QinetiQ and Thales to demonstrate this set up at the Hell Bay 4 trials at Unmanned Warrior in October 2016, where the focus on optimizing missions for the assets available was clearly demonstrated. Over-horizon-operations depend largely on reliable and effective communications between the command and control centre and between the vehicles themselves. In order to make this happen a relay surface or air vehicle is used. By running communications through an aerial drone, this relay link meant that the surface and subsea vehicles were able to operate at a far greater distance from the shore.


10 Unmanned Vehicles Networked Through Neptune for Unmanned Warrior 2016 (Courtesy of Royal Navy)

This 'best of breed' approach effectively allows goal-based missions to benefit from the best attributes from each vehicle type. This crux of this approach is to use a network of unmanned systems to bring together the best attributes of each vehicle type. Whether that may be an aerial drone, surface vessel or unmanned underwater vehicle, the ultimate goal is to have the ability to adapt the make-up of the fleet to create a robust system that is reliable enough to be deployed for even the most demanding circumstances such as EOD operations. Essentially it allows the fleet to benefit from the exceptional positioning communications feed of an aerial drone, the endurance of USV, along with the AUVs on the front line.

Together with ASV Global, Bluebear, and QinetiQ the team were successful in facilitating the collaboration of unmanned vehicles including air, surface, and subsea, on common missions running through Dstl's Maritime Autonomy Framework (MAF) realised through SeeByte's Neptune software. Using vehicles from Hydroid, OceanServer Technology, SeaRobotics, Bluebear and ASV Global, the team networked 10 unmanned systems, from three different countries through a single command and control station.



10-13-2016 12:16: ETA: 10-13-2016 13:50:37 0
Duration: 0 h : 6 m : 44 s ETR: 1 h : 27 m : 6 s AO: 1 / 5
Complete: (0.0) %

Name	Instance	Phase	State	Progre..	Time	Color	Age	Comms
MER3...	Survey3	SURVEY	TASK	7%	0 h : 41...	🟪	6 m : 4...	✗
MER3...	Survey5	SURVEY	TRANS...	[illegible]	0 h : 0 m	🟦	5 m : 3...	✗
SN014...	No obj...	NONE	NONE	0%	-	🟧	10 m : ...	✗
USV26...	No obj...	NONE	NONE	0%	-	🟪	8 m : 5...	✗

Parameter	Value
North	329.536
East	1109.686
Depth	0.00
Altitude	0.00
Roll	0.00
Pitch	0.00
Yaw	331.765

Command and Control of 10 Unmanned Maritime Systems at Unmanned Warrior 2016

6 CONCLUSIONS

Going forward, there are several hurdles that must be overcome before these advancements in technology become viable solutions for modern warfare. Scheduling UMS can be challenging where operational areas are beyond a single UMS capability in a single mission or where multiple assets are available with varying capability.

SeeTrack V4 allows users to create Q-Routes or polygons which can encompass an area greater than the maximum surveyable area of a UxV. If a survey of the area is to be executed, the larger area must be divided into smaller pieces. SeeByte have developed tools to automate this process, and allows the user to specify division criteria in order to speed up operation.

A schedule management system has been designed that will allow operators to organize the execution of large unmanned maritime operations consisting of multiple smaller areas. Using this, an operator is able to input things like waterspace and asset availability or time constraints and get the best (shortest total operation time) way to run the whole operation. The system is then able to automatically generate a plan for multiple assets based upon their capability and availability.

SeeByte Software Solutions

SeeTrack is the leading technology used by Explosive Ordnance Disposal (EOD) and Clearance Divers around the world, to achieve success in managing their off-board assets during dangerous and demanding missions.

SeeByte has created a variety of product offerings to help manage MCM assets, ultimately providing situational awareness across all assets and within the battlespace for MCM and EOD Operators.

For more information please contact SeeByte at sales@seebyte.com