Chapter 10 – PT-1: SUPERVISORY CONTROL: OPTIMAL DISTRIBUTION OF WORKLOAD AMONG OPERATORS FOR MIXED INITIATIVE CONTROL OF MULTIPLE UAVs

Gil Gonçalves, Rui Gonçalves, Sérgio Ferreira and José Pinto
Department of Informatics Engineering
Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias s/n
4200-465 Porto
PORTUGAL

Email: gil@fe.up.pt

João Sousa
Department of Electrical and Computer Engineering
Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias s/n
4200-465 Porto
PORTUGAL

10.1 DATES

24 January 2011 – 4 February 2011.

10.2 LOCATION

OTA, Lisbon, Portugal.

10.3 SCENARIO/TASKS

At the Underwater Systems and Technology Laboratory (LSTS) [1] we have been designing, building and operating a number of heterogeneous unmanned vehicles. These include Remotely Operated Vehicles (ROV) [2], Autonomous Underwater Vehicles (AUV) [3],[5],[6], and Autonomous Surface Vehicles (ASV) [4]. We have been also developing UAVs [7] as a result of our collaboration with the Portuguese Air Force Academy.

Recent technological advances led to the creation of very capable unmanned systems constructed using low cost hardware. This allows the application of these technologies to scenarios where multiple unmanned systems can be employed simultaneously like patrolling, adaptive sensing, search and rescue, etc. However, human operators have turned into an increasingly scarcer and more expensive resource whose exploitation shall be optimized.

In this chapter, we describe a conceptual framework for optimal inclusion of the operator in the control loop and the application of these concepts into a Command and Control (C2) operator interface. Our objective is to distribute and reduce the workload of a decentralized team of operators controlling multiple UAVs. To achieve this goal we intend to advise operator's actions and reconfigure C2's layout using an automated methodology. The operator can have different levels of situation awareness, at different stages of the mission. The system will help operators to dynamically configure an optimal view of the mission state from a set of predefined console layout profiles.

We interpreted and adapted the original (Level Of Autonomy) LOA matrix (Table 10-1) into our framework for optimal inclusion of the operator in the control loop. The LOA-Level of Autonomy Table [10] is based on

Sheridan's 10-level of autonomy scale [11] and simplified to present only eight levels of autonomy. The two dimensions of the matrix (Table 10-1) are the eight levels (matrix rows) crossed with four functional categories (matrix columns). The second dimension presented in this matrix is the division of each task into four functional steps. These tasks present human decision-making processes as a set of OODA cycles (Observe, Orient, Decide, and Act).

Table 10-1: Partial LOA Matrix as Originally Published in [10].

Level	Observe	Orient	Decide	Action
8	The computer gathers, filters, and prioritizes data without displaying any information to the human.	The computer predicts, interprets, and integrates data into a result which is not displayed to the human.	The computer performs ranking tasks. The computer performs final ranking, but does not display results to the human.	Computer executes automatically and does not allow any human interaction.
7	The computer gathers, filters, and prioritizes data without displaying any information to the human. Though, a "program functioning" flag is displayed.	The computer analyses, predicts, interprets, and integrates data into a result which is only displayed to the human if result fits programmed context.	The computer performs ranking tasks. The computer performs final ranking and displays a reduced set of ranked options. Without displaying "why".	Computer executes automatically and only informs the human if required by context. It allows for override ability after execution. Human is for shadow contingencies.
		…		
		…		
1	Human is the only source for gathering and monitoring (defined as filtering and prioritizing) all data.	Human is responsible for analyzing all data, making predictions and interpretation of the data.	The automate does not assist in or perform ranking tasks. Human must do it all.	Human alone can execute decision.

Table 10-2 is used to categorize the operator skills using the LOAs he is certified to respond to, the CP (Console Profile) the operator is familiarized and the number of vehicles he can handle safety at a certain LOA.

Table 10-2: Fields Used to Infer About the Operators Skills in the Framework.

Certified Type of LOA	Certified Consoles Profiles	Number of Vehicles
Type of manoeuvre the operator is certified.	Set of operation Consoles the operator is familiarized. By preference order. (for one LOA)	Operator fan-out of vehicles (for one LOA)

To exemplify the framework's execution we will evaluate a mission scenario where the operators have to find a target and follow it. There will be two operators and five UAVs in this scenario.

Currently existing UAVs offer little adaptability in terms of automation: operators can command the UAV to fly autonomously, following a pre-defined flight path, or they can control it manually. For this example we will use 2 LOAs for the operators, and another one of full autonomy used in handover and in emergency situations. The operators LOAs to be used are further sub-divided into a high level control LOA and low level control LOA in this scenario.

All three LOAs used are described as follows:

- **Operational Mode 1 –** Tele-Operation or Direct Control – LOA = (3,2,2,2);
- **Operational Mode 2 –** Survey – LOA = (6,6,7,6); and
- **Operational Mode 3 –** Full Autonomy – LOA = (8,8,8,8).

The matrix represented in Table 10-1 can be related with the creation of different types of console profiles. Different console profiles can be associated to different combinations of the four functional categories (OODA) – operational modes. For the presented framework we have a direct relation of LOA and CP. The formal representation for CP-LOA tuple is:

$$\text{CP-LOA} = (\{Obs_1 \ldots Obs_n\},\{Ori_1 \ldots Ori_n\},\{ Dec_1 \ldots Dec_n\}, \{Act_1 \ldots Act_n \})$$

The elements on the tuple are represented as sets so we can group the OODA functional categories. This way it is possible to have one CP capable of handling different Operational Modes.

We will use two CPs (CP1 = ({3},{2},{2},{2}) and CP2 = ({6-7},{6-7},{6-7},{6-7})) to handle this mission example as follow:

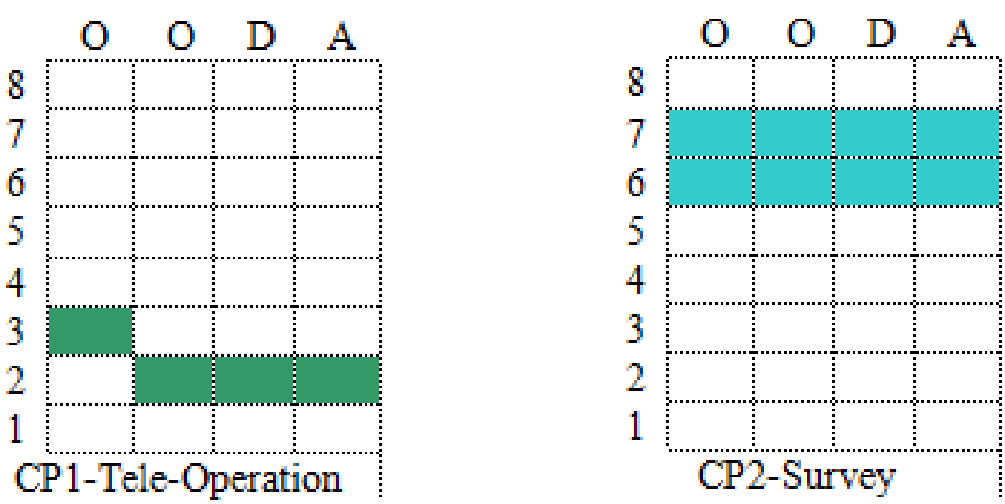

Figure 10-1: Two Console Profiles Used in Mission (For Low and High Level Control).

For this mission example we will have two operators with the following Skills Tables.

Table 10-3: Skills Table – Operator 1 Can Handle 3 UAVs in High Level Control and 1 UAV in Low Level Control; Operator 2 Can Handle 4 UAVs in High Level Control.

	Certified Type of LOA	Certified CPs (Consoles Profiles)	Number of Vehicles
Operator 1	(3,2,2,2)	{CP1}	1
	(6,6,7,6)	{CP2}	3
Operator 2	(6,6,7,6)	[CP2}	4

Figure 10-2 illustrates the 5 most important steps taken when one of the operators finds the target. The state of the system before any of the operators finds the target is the beginning step (step 1) of Figure 10-2. Initially, all the UAVs are in survey mode – mode 2 of our LOA definition. Both of the operators are using CP2 to control the UAVs: define survey areas and look at part of the payload data (video).

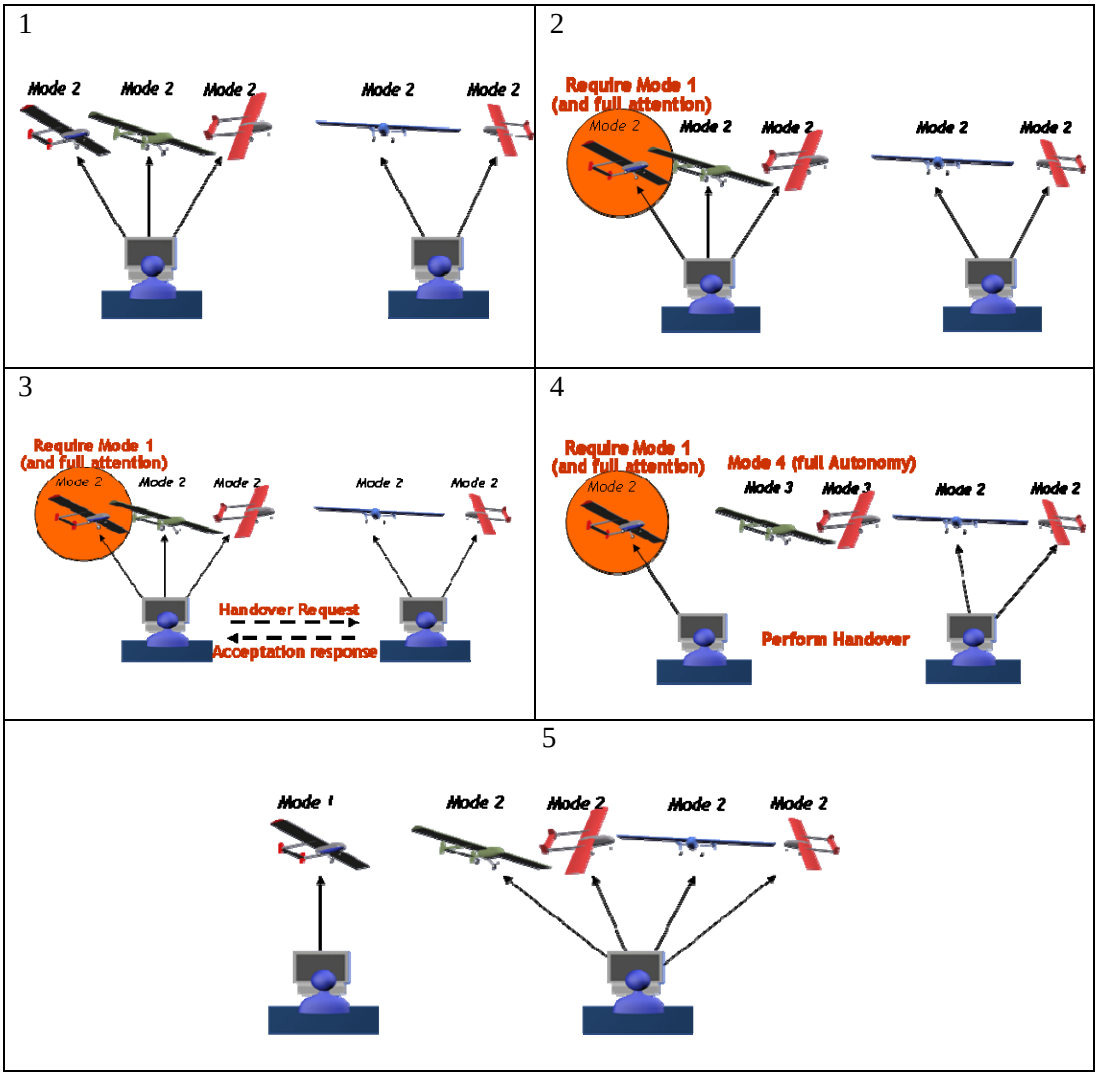

Figure 10-2: Logic of Operation for Workload Distribution Mission Example.

In step 2 of Figure 10-2, Operator 1 finds the target. The target must be followed using direct control. To solve the excessive workload of Operator 1 (Operator 1 can handle only 1 UAV in Operational Mode 1 – Tele-operation – and Operator 2 is not certified for Operational Mode 1, consult Table 10-3), the system (mission supervisor) will try to assign this UAV in mode 1 – Tele-Operation – to some operator. The only operator capable of handling mode 1 is operator 1, as defined in Table 10-3. Since the operator 1 is capable of handling only one UAV in this Mode, the mission supervisor will advise Operator 1 to hand-over the other 2 UAVs from Operator 1 to Operator 2. Here starts step 3 with the handover process: Operator 1 releases the two controlled UAVs by setting them at mode 3 (Full Autonomy).

Finally, in step 5, Operator 2, that has accepted the hand-hover, takes over these UAVs which are on Mode 2 and the Operator 1 can now handle Mode 1 (Tele-Operation) and follow the target. In this step the Mission Supervisor advises Operator 1 to use CP1-Tele-operation to respond mode 1 LOA, which requires full attention to the vehicle, according to his skills in Table 10-3.

10.4 TECHNOLOGIES EXPLORED

The concepts of operation for multi-UAV teams differ from single UAVs in the sense that in the former there exist common objectives like maintaining a common knowledge database [8] and redundant execution of crucial actions [9].

In our C2 framework, UAVs can be tasked either individually by an operator or they can be tasked by a software agent that acts as an operator (Team Supervisor). The team supervisor divides work among the vehicles according to a multi-UAV mission specification and simple task-allocation algorithms. If the control over the UAV is not overridden, they carry out planned behavior until they are faced with failures, or there are any other unpredicted situations in which they contact the ground station and require human intervention.

To provide system-level control of multiple vehicles, we use a software agent that holds a multi-UAV mission specification. This mission specification is currently a list of individual plans that need to be executed by UAVs. Tasks are divided among UAVs in a way that workload is shared among capable vehicles. Some tasks however also require the intervention of human operators for correct execution, so the availability of operators must be taken into account by the team supervisor while tasking the network.

As stated before, this framework was employed in an existing C2 software framework: Neptus. Neptus has an underlining architecture that provides the means for creating the various consoles used in different CPs. This section introduces Neptus and gives an example of such consoles.

Neptus is a distributed C2 framework for operations with networked vehicles, systems, and human operators. Neptus supports all the phases of a mission's life cycle: planning, simulation, execution, and post-mission analysis. Neptus supports concurrent operations. Vehicles, operators, and operator consoles come and go. Operators are able to plan and supervise missions concurrently [12].

In Neptus, the Console Builder application facilitates the addition of new vehicles with new sensor suites to Neptus. In this application the operator can build, configure, and save vehicle consoles. There are two important aspects for console configuration: visual components and event communications. The internal Neptus event communication system is based on a tree structure (following the blackboard design pattern [13]), where nodes indicate the subject of data values in leafs. Neptus visual components can become listeners of a single variable (tree leaf) or of a defined variable domain (tree branch). Whenever a message arrives, using the IMC [14] communication protocol, that data is stored in a specific tree branch and listeners for any

branch that encloses the affected branch/leaf are informed of the incoming network data. In a similar way, output data is sent to the network by Neptus console components through the variable tree. The variable tree system is also used for event communication between Neptus console components.

There are two states in the Neptus generic console builder application: editing and operational. In editing mode, the palette of available components (STANAG planning panel, compass panel, renderer panel, video panel) becomes visible. Users can then add and place components freely inside console main panel. Component properties can be edited to connect the panels to different systems and variables. When all components are ready, correctly placed and connected to the system variable tree, the user can switch the state of the application to the Operational mode. In this mode, the position of the components in the console is fixed and it responds to the user interactions (Figure 10-3).

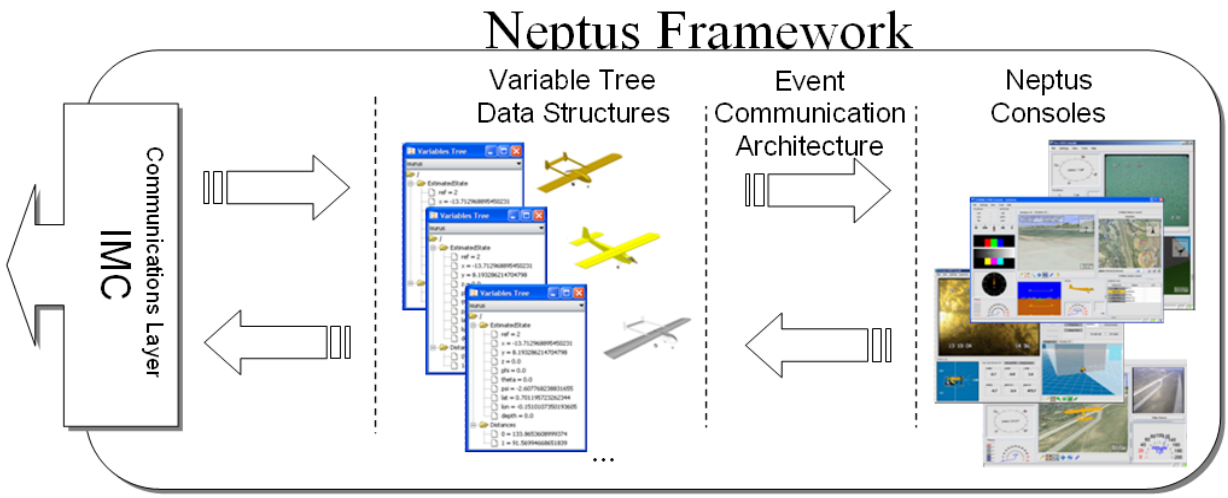

Figure 10-3: Neptus Internal Communications System.

Besides having the capability of dynamically creating new consoles during a specific mission, Neptus also has predefined consoles already available for the LOA switches the presented framework requires. These consoles go from standard tele-operation consoles, as seen as example 1 of Figure 10-4, to supervision consoles, as seen on the right (example 2) of Figure 10-4. These consoles have different layouts depending on the central function they have. For instance a tele-operation console will typically have more detailed data about the UAV under its control, whereas a supervision console will only have a simplified view of the current UAV to allow a broader view of the whole team. As an example of said consoles we introduce the details behind the current flight manager console used for UAV mission supervision at the LSTS.

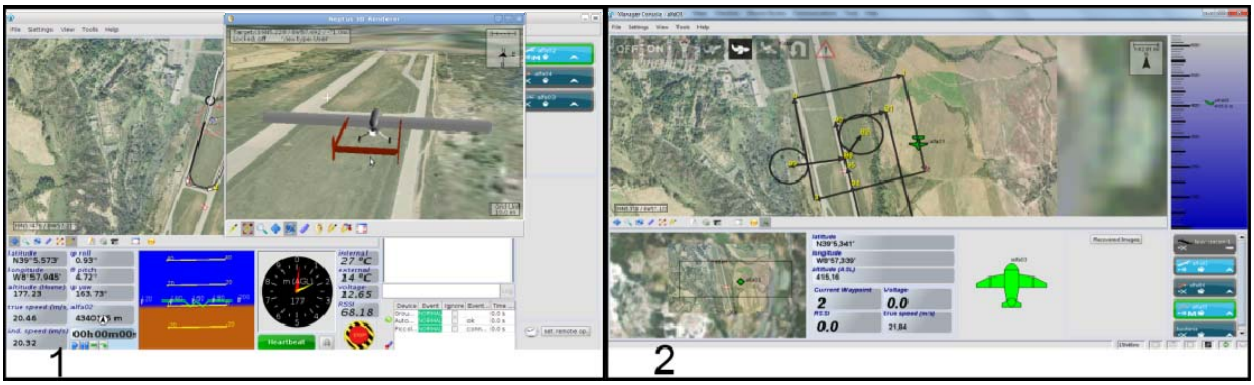

Figure 10-4: Tele-Operation (Low Level Control – 1) and Supervisory Control (High Level Control – 2) Consoles.

The supervisory control console, as seen in Figure 10-4, was developed based on a Real-Time Strategy (RTS) paradigm with the intent of applying the concepts, learned by this type of games, on how to efficiently control and supervise groups of units of various dimensions and with varying capabilities. This approach, while not being new, has allowed the implementation of a console which supports high LOA levels CP-LOA = ({6-7},{6-7}, {6-7}, {6-7}) while, at the same time, enables the supervision of UAV teams with a low workload rating value for the operators.

10.5 HUMAN FACTORS ISSUES EXPLORED

The two main human factors that we explore in this framework is situation awareness and workload index. To that extent, we perform systematic evaluations of these metrics through the use of NASA's TLX method, for workload analysis and the SAGAT method for situation awareness testing. An example of the workload values collected in one of these tests can be viewed in Figure 10-5.

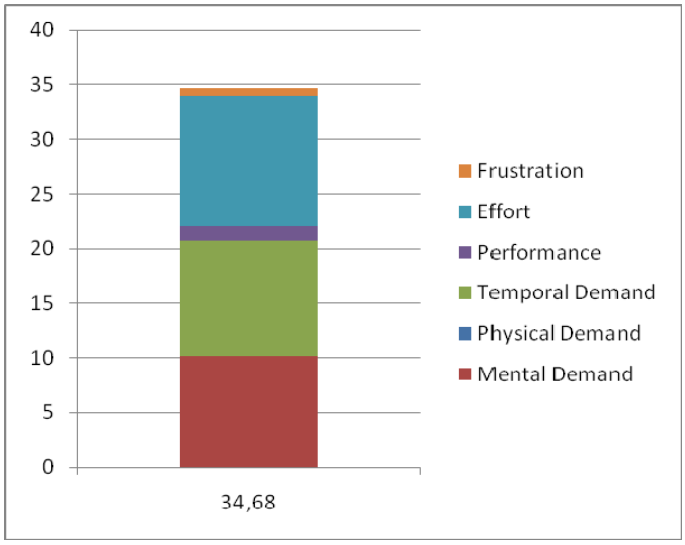

Figure 10-5: Flight Manager Console's Total Workload Rating, Using NASA-TLX [15], in a 3 UAV Scenario.

10.6 UNMANNED SYSTEMS USED

For the various flights performed a vast array of unmanned systems were used. Some of these systems can be viewed in Figure 10-6.

Figure 10-6: UAVs ANTEX X02 1-4 Series.

Due to maintenance reasons, it is for us very common to change UAV models in the middle of a test run. Nevertheless we present specific details of the systems which are more regularly used on the shakedown tests.

Table 10-4: Specific Data of the ANTEX X02 – 03 UAVs.

	ANTEX X02 Series
Max Weight	10 KG
Width	2,4 m
Max Speed	150 Km/h
Max Payload	4 Kg
Max Autonomy	5 h
Max Altitude	2 Km

10.7 SUMMARY OF ANY NATO COMMUNICATIONS/COLLABORATIONS/ INTERACTIONS

	Planning/Design	***Execution***	***Analysis***
Communication	*X*	*X*	*X*
Coordination			
Collaboration			

10.8 SUMMARY OF TD RESULTS

We presented the concepts behind a framework for managing UAV task and workload allocation between various operators in a mission scenario. This framework was applied to the development of a Command and Control (C2) application which is capable of self-adaptation, operator advisement and automatic task distribution among operators and UAVs according to mission objectives, phase and occurrences. An example scenario of this framework, as well as an example of the details around one of the consoles used by the operators, was presented and discussed.

10.9 LESSONS LEARNED

This C2 application enables a clear view and presence on the remote environment by putting the operator much closer to the control loop, whether it is high level or low level control, with the consequent improved redistribution of tasks and situational awareness. NASA Task Load Index (TLX) was used as a means to determine the adequacy of the C2 interface and functionalities. The preliminary results obtained with this framework are promising and we are confident that its use will vastly improve the reliability of multi-UAV teams by augmenting their compatibility with more mission scenarios.

10.10 STUDY CONSTRAINTS/LIMITATIONS

Although we have a vast base of operations and a large array of working vehicles to test this framework, the majority of the testing took place in a simulated environment. This will, on one hand, allow us to detect faults in the system without endangering our vehicles but, on the other hand, limit the conclusions that can be extrapolated from the experiments due to lack of realism.

10.11 CONCLUSIONS

Throughout this chapter we referenced the growing importance of multi-UAV systems, paying special attention to the need of optimal in-the-loop inclusion of operators for the successful use of these systems. The tested framework for managing UAV task and workload allocation between various operators in a mission scenario proved to be an improvement over the previous approach. Further testing, with simulated scenarios, will provide new insights over the real capabilities of the proposed framework.

10.12 FUTURE RESEARCH NEEDS AND PLANS IN THIS AREA

We intend to implement another formal layer over the presented framework to extract the viability of mission execution. It is possible to reach combinations of plan state manoeuvres that overload the response of the operator's team. Our approach will use Petri Nets for the model to tackle this issue (another similar example can be consulted in [16]). By studding the plan loaded in each UAV and applying a transformation that combines all UAV plan states, the plan state change events probabilities, and the operators team recourses into a Petri Net, we can infer about the probability of reaching a failure state. The failure state can be considered to be a state where operator resources do not correspond to the mission state demands. In the last analysis, we can know the probability of reaching one mission state before the Mission Team Supervisor has to process the resource allocation. This information can be used to optimize the resource allocation process and also to help avoiding some mission states in the mission planning phase (e.g., find and avoid states that require full autonomy LOA = (8,8,8,8) manoeuvres).

10.13 ACKNOWLEDGEMENTS

The authors acknowledge the contribution of AsasF-FEUP team in the requirements identification and in the demonstration tests. Also acknowledge the Portuguese Air Force Academy (AFA) in the field tests and developments and design of ANTEXs X02 UAVs under the *PITVANT* project funded by the Portuguese Ministry of Defence. Neptus relies on other components of the LSTS tool chain for its correct execution. For this, we would like to thank Eduardo Marques, Paulo Dias, and Ricardo Martins mainly for their efforts in developing the IMC protocol. Rui Gonçalves and José Pinto were partially supported by FCT grants.

10.14 REFERENCES

[1] USTL. Underwater Systems and Technology Laboratory, February 2011. http://whale.fe.up.pt.

[2] Gomes, R., Sousa, A., Fraga, S.L., Martins, A., Borges Sousa, J., Lobo Pereira, F. et al. A New ROV Design: Issues on Low Drag and Mechanical Symmetry. In Oceans05 Europe, June 20-23 2005.

[3] Cruz, N., Matos, A., Borges de Sousa, J., Lobo Pereira, F., Estrela Silva, J. Coimbra, J. and Brogueira Dias, E. Operations with Multiple Autonomous Underwater Vehicles: The PISCIS Project. AINS, 2003.

[4] Ferreira, H., Martins, R., Marques, E. et al. Swordfish: an Autonomous Surface Vehicle for Network Centric Operations. In IEEE Oceans Europe, 2007.

[5] Madureira, L., Sousa, A., Sousa, J. et al. Low Cost Autonomous Underwater Vehicles for New Concepts of Coastal Field Studies. In CERF ICS, 2009.

[6] USTL. Seascout, January 2010. http://whale.fe.up.pt/seascout.

[7] Pereira, E., Bencatel, R., Correia, J. et al. Unmanned Air Vehicles for Coastal and Environmental Research. In CERF ICS, 2009.

[8] Jariyasunant, J., Pereira, E., Zennaro, M., Hedrick, K., Kirsch, C. and Sengupta, R. "CSL: A Language to Specify and Re-Specify Mobile Sensor Network Behaviors", in Proceedings of RTAS 2009, 2009.

[9] Borges Sousa, J., Simsek, T. and Varaya, P. "Task planning and execution for UAV teams", in Proceedings of CDC'04, 2004.

[10] Proud, R.W., Hart, J.J. and Mrozinski, R.B. Methods for determining the level of autonomy to design into a human spaceflight vehicle: A function specific approach. In Performance Metrics for Intelligent Systems Workshop, Gaithersburg, MD (2003).

[11] Sheridan, T.B. Telerobotics, automation, and human supervisory control. MIT Press, Cambridge, MA, USA (1992).

[12] Sousa Dias, P., Pinto, J., Gonçalves, R., Borges Sousa, J., Pereira, F.L. and Gonçalves, G. Neptus, command and control infrastructure for heterogeneous teams of autonomous vehicles. In International Conference on Robotics and Automation. ICRA2007, IEEE, April 2007.

[13] Deugo, D., Weiss, M. and Kendall, E. Reusable patterns for agent coordination. In: Omicini, A., Coordination of Internet Agents, pp. 347-368. Springer, 2001.

[14] Martins, R., Sousa Dias, P., Marques, E.R.B., Pinto, J., Sousa, J.B. and Pereira, F.L. Imc: A communication protocol for networked vehicles and sensors, Oceans, 2009.

[15] Hart, S.G and Staveland, L.E. Development of a multi-dimensional workload rating scale: Results of empirical and theoretical research. In P.A. Hancock and N. Meshkati (Eds.), Human Mental Workload. Amsterdam, 1988.

[16] King, J., Pretty, R.K. and Gosine, R.G. Coordinated execution of tasks in a multiagent environment. IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, Vol. 33, No. 5, September 2003.