Autonomic Management for the Next Generation of Autonomous Underwater Vehicles

Carlos C. Insaurralde
Ocean Systems Laboratory
Institute of Sensors, Signals and Systems
Heriot-Watt University
Edinburgh, Scotland, UK
c.c.insaurralde@hw.ac.uk

***Abstract*—Underwater vehicles are increasing their autonomous capabilities more and more in order to carry out more complex and longer missions. This basically requires operational resilience and efficient energy consumption to succeed in persistent presence. Some of current Autonomous Underwater Vehicles (AUVs) have a large degree of self-governance but most of them lack self-management (e.g. auto-maintenance before, during, and after missions). This paper introduces the autonomic computing concept to AUV control architectures in order to explore a solution that endows AUVs with resilience and greenness. One of the attractive characteristics of self-managed AUVs is that automatic functions from self-managing capabilities are executed in background, i.e. the deliberative control layer gets rid of tasks that are now placed in the reactive one. This paper also presents a review of the approaches for self-managed systems, a discussion on suitability of current autonomic technologies, and future research directions.**

***Index Terms*—Autonomic Management, Autonomous Underwater Vehicles, Physiologically-Inspired Resilience, Oceanic Engineering.**

I. Introduction

Underwater vehicles are increasing their autonomous capabilities more and more in order to carry out more complex and longer missions, e.g. very big and hostile exploration areas in seafloor surveys. This basically requires operational resilience and efficient energy consumption to succeed in persistent presence.

Current Autonomous Underwater Vehicles (AUVs) are self-governed, some of them have a large degree of self-governance but most of the approaches lack self-management. Autonomic Computing (AC) systems essentially provide the following self-managing capabilities: self-healing, self-protecting, self-optimizing, and self-configuring. They come from a biological metaphor based on the self-regulating capabilities of the autonomic nervous system in the human body.

This paper introduces the AC concept to AUV control architectures to endow them with resilience and greenness. Self-healing means diagnosis and mitigation of faults so that reliability and consequently availability can be guaranteed. Self-optimizing means tune-up and continuous monitoring of system capabilities so that resource allocations and workloads can meet requirements as well as efficiency and effectiveness. Self-configuring means automatic reflection of, and adaptation to the dynamically changing environments. Self-protecting means anticipation, detection, and identification from internal and external attacks so that security can be assured at any time. The above capabilities are fundamental to endure the persistent presence of AUVs in complex and long missions. Additionally, they provide AUVs with self-aware, self-adjusted, and self-situated abilities. One of the attractive characteristics of self-managed AUVs is that automatic functions from the above capabilities are executed in background, i.e. the deliberative control layer gets rid of tasks that are now placed in the reactive one.

This paper presents the architectural aspects, and details of design and realization of a AC-enabled AUV control architecture. It analyses potential software platforms as candidates for the system implementation/integration, and identifies control algorithms by addressing different strategic aspects such as adaptation, robustness, intelligence, optimization, and hierarchy.

The rest of the paper is structured as follows. Section II presents the background of this research work by reviewing fundamental concepts for autonomic management. Section III presents a review of relevant approaches. Section IV presents self-* aspects of a AUV control architecture based on self-managing capabilities. Section V describes the autonomic management for AUVs. Section VI discusses the assessment process proposed based on the results obtained. Last Sections are devoted to the conclusions, and future direction for this research.

II. Background

This Section introduces the foundations and principles of the AC paradigm. The computational principle is inspired by the self-managing ability provided by the Autonomic Nervous System (ANS) of the human body.

A. Autonomic Nervous System

The Autonomic Nervous System (ANS) controls the involuntary activities of the organ muscles, and glands by means of sensory (afferent), and motor (efferent) neurons. The

ANS receptors (sensors) are located in the plasma membrane of certain cells, and ANS transmitters (effectors) are associated with the viscera (internal organs). The cells are able to respond by sending a signal over the connection defined by the sensory-motor neurons. The cell response can be excitatory or inhibitory, depending on the cell type.

The ANS plays a role as the housekeeper in the human body by making use of above neural linkage. The ANS influences the visceral activities by maintaining a relatively stable internal environment, and making adjustments to keep optimal conditions. The autonomic control effects are fast, and essential for homeostasis[1] [1]. An example of homeostatic mechanism is the thermoregulation is the ability of a living system to keep its body temperature within certain boundaries, even when the surrounding temperature changes.

Homeostatic adaptation requires coordination between actions on the external world (surroundings) and activities of the internal environment. Figure 1 shows the above contextual linkage. This figure also includes withdrawal reflexes (on the right) that are not classified as part of the ANS but interesting to be taken into account. There are other reflexes in the peripheral nervous system such as stretch reflexes that are also relevant to the for AUV control architecture studied.

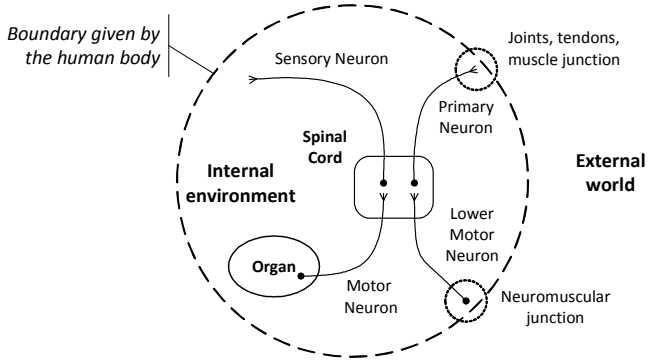

Fig. 1. *Internal and external contexts.*

The ANS is divided into three functional parts: sympathetic, parasympathetic, and enteric divisions. The sympathetic division stimulates those activities that are mobilized by the organism during emergency and stress situations. The parasympathetic division stimulates activities associated with conservation and restoration of body resources. The enteric division is an intrinsic network of neurons and connections based on the combination of sympathetic and parasympathetic neurons [1].

The sympathetic and parasympathetic divisions have both structural and functional differences. They normally work in an opposing manner (stimulatory and inhibitory effects) by restoring balance of stable conditions to maintain homeostasis.

Table 1 lists the response of heart to sympathetic and parasympathetic stimulation. The diversity of autonomic responses and functions is achieved primarily by different types of receptors.

[1] Term proposed by Claude Bernard, a French physiologist, to define the property of a system that regulates its internal environment and tends to maintain its stable condition regarding the external environment.

TABLE I. HEART SYMPATHETIC AND PARASYMPATHETIC FUNCTIONS

Organ	Target	Sympathetic effect	Parasympathetic effect
Heart	Heart rate	Increase	Decrease
	Heart force of ventricular contraction	Increase	Decrease
	Deep coronary arteries	Vasodilation	Slight vasodilation
		Vasoconstriction	No effect
	Blood vessels of most viscera	Vasoconstriction	Vasodilation
	Blood vessels of skeletal muscles	Vasodilation	No effect
	Blood vessels of skin	Vasoconstriction	Vasodilation, blushing
	Platelets (blood clotting)	Increased clotting	No effect

The regulation of the ANS is carried out by autonomic reflexes, although the cerebrum, the hypothalamus, and other areas of the brain have influence on the autonomic functions. The ability to maintain homeostasis is limited without ANS regulation. An example of combined autonomic reflex is the self-regulation of the cardiovascular functions (Figure 2).

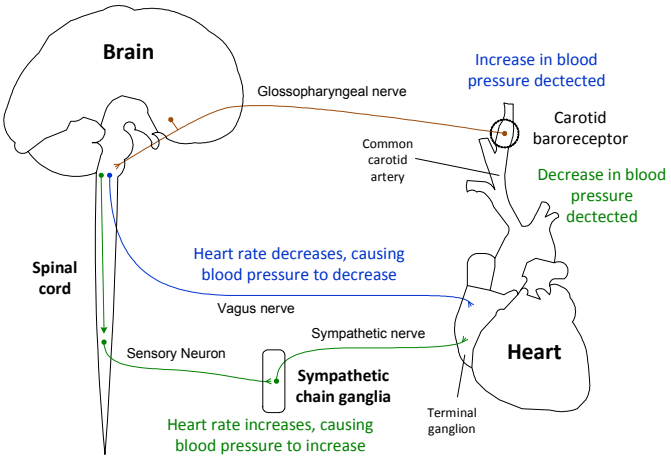

Fig. 2. Blood pressure reflex.

Baroreceptors placed in the walls of large arteries (near the heart) detect changes in blood pressure. Sensory neurons transmit information from the baroreceptors through the glossopharyngeal and vagus nerves to the medulla oblongata. Interneurons in the medulla oblongata integrate the information, and action potentials are produced in autonomic neurons that extend to the heart. If baroreceptors detect an increase in blood pressure, parasympathetic reflexes decrease heart rate, returning blood pressure back to normal. On the contrary, a sudden decrease in blood pressure initiates a sympathetic reflex which stimulates the heart to increase its rate and force of contraction in order to increasing blood pressure [2].

B. Autonomic Computing

The AC paradigm was proposed by IBM for management of Information Technology (IT) resources in response to the demand of a computer system that would foresee the users' needs. This solution basically helps IT companies eliminate the increasing costs of restoring hardware and software failures so more reliable and dependable systems can be developed. AC effectively prevents downtimes and system failures [3].

AC was conceived to lessen the spiraling demands for skilled IT resources, reduce complexity, and better exploitation of its potential to support higher order thinking and decision making [4]. The AC philosophy is to design and build computing systems capable of running themselves, adjusting to varying circumstances, and preparing their resources to handle most effectively the workloads. These ACs anticipates needs and allows users to concentrate on they want to accomplish rather than figuring how to rig the computing systems [5].

The AC paradigm is inspired by the ANS functionality. It comes from a biological metaphor based on the self-regulating capabilities of the ANS in the human body. AC systems essentially involve the following self-managing capabilities:

- **Self-healing** means diagnosis and mitigation of faults so that reliability and consequently availability can be guaranteed.
- **Self-optimizing** means tune-up and continuous monitoring of system capabilities so that resource allocations and workloads can meet requirements as well as efficiency and effectiveness.
- **Self-configuring** means automatic reflection of, and adaptation to the dynamically changing environments.
- **Self-protecting** means anticipation, detection, and identification from internal and external attacks so that security can be assured at any time.

Additionally, the above capabilities provide systems with self-aware, self-adjusted, and self-situated abilities.

From a control viewpoint, AC is a closed-loop control system. It has the typical couple: controlling and controlled devices. The former device is the controller which is in charge of making decisions to control the latter one, i.e. the managed resource. The feedback is achieved by measuring on the controlled device. Figure 3 shows the closed-loop AC control.

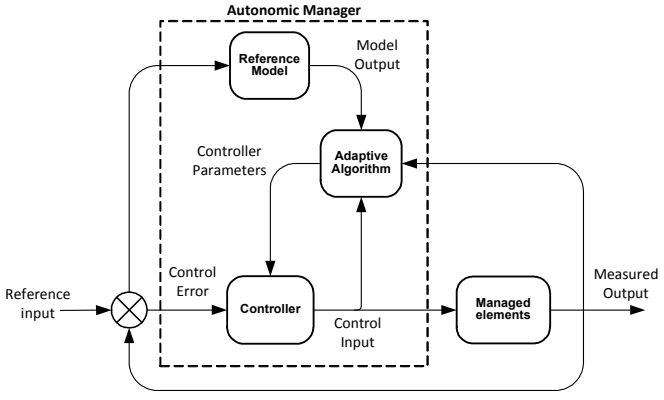

Fig. 3. Closed-loop control for AC.

Thus, the AC architecture involves autonomic elements built of an autonomic Manager (AM), and a managed element (ME) or resource. The former is the AC controller in the closed-loop control. The latter is the controlled system.

The decision-making cycle of the AM is an OODA[2]-compliant process. It is given by the following steps: Monitor, Analyze, Plan, and Execute (MAPE). The autonomic element manages its own internal state and its interactions with its environment.

Figure 4 shows the anatomy of autonomic elements, i.e. AM plus Managed Element (ME).

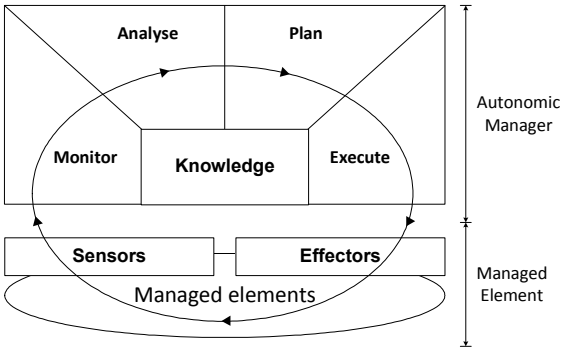

Fig. 4. Autonomic elements.

The monitor observes the sensors, filters the data collected from them, and then stores the distilled data in the knowledge base. The analysis engine compares the collected data against the desired sensor values also stored in the knowledge base. The planning engine devises strategies to correct the trends identified by the planning engine. The execution engine adjusts parameters of the ME by means of effectors, and stores the affected values in the knowledge base [7].

The big challenge for the design of AM is the rules, i.e. AC policies that govern the transitions to go from a current state to new one. Tackling this issue, three types of AC policies at different levels of abstraction are distinguished: action, goal, and utility-function policies [8].

III. Literature Review

The literature review presented in this paper has a broader focus since AC applications for AUVs are difficult to be found. However, there is a proposal to implement autonomic elements for AUV architectures [9]. This approach purposes to merge the autonomic management into the autonomous controller, and deploy it for each system module. Likewise, survivability is in the thick of AUV researches by addressing reduction of susceptibility and vulnerability [10], e.g. reduction of susceptibility through collision rules [11].

The state-of-the-art remainder focusses on general robotics with a strong point in mobile robots.

The AC paradigm is gaining interest in the robotics community but no many proposals can be found. The AC concept was introduced more than a decade ago as an IT solution.

[2] Observe-Orient-Decide-Act decision-making loop [6].

A Monograph from the National Aeronautics and Space Administration (NASA) presents the steady progress of the technologies enabling autonomous and autonomic behaviors of spacecraft [12]. It argues for the need of a much farther technological extension to enable success in the most advanced un-crewed space missions in the future. It describes these technologies and their relevance not only for space missions of the NASA but also for advanced future mission concepts. This literature identifies the need of a combination of automatic, autonomous, and autonomic systems in order to reduce costs of missions but fails to show which and how autonomic capabilities are realized in spacecraft.

A pioneering research work proposes mobile robots as an excellent test-bed for research on AC. It recognizes the self-management power by exploring the use of AC techniques in the domain of ground-based mobile robots [13]. The main focus is on robustness and fault-tolerance, assuming that all hardware eventually fails, making control decisions based on bad sensor data is very dangerous, and software written on-site is often very buggy. This research work only presents the ideas to apply AC to mobile robots but no any implementation.

Other research works investigate particular self-* capabilities, and mention AC as a potential effective solution. For example, self-healing (including self-reproduction) for swarm robotics systems [14]. This research proposes a swarm robotic system architecture based on virtual neurons, autonomous self-diagnosis, consequence-oriented prescription, autonomous self-curing, and self-reproduction. The self-healing capability automatically detects errors, and makes systems recover from failures. Another example dealing with self-healing capabilities from AC is the autonomic behavior-based software architecture for mobile manipulator [15].

ADE infrastructure inherently enhances the "intelligence" of robotic architectures by means of AC mechanisms. It provides the intertwined features of system start-up, failure detection, failure recovery, and dynamic system reconfiguration, with no extra effort from the designer [16].

A model-based utility-driven approach to the autonomic management of mobile robot resources is an effective solution to use mobile robot resources. This approach builds an autonomic solution capable of optimizing a utility function supplied by the robot administrator. This self-optimization involves adapting the operation mode of the robot components to changes in the amount of resources available, and in the environment [17].

Self-organized distribution of tasks is proposed for autonomous mobile robotic systems. An autonomic self-manager distributes the needed tasks in an optimal manner by keeping their minimal execution times for the robotic actions. The self-manager tries to be dynamically adaptive [18].

IV. Autonomic Control Architecture

This Section presents an AC approach to support self-management in AUV control architectures.

A. Robotic Architecture

Figure 5 shows the AUV control architecture proposed that includes autonomic management.

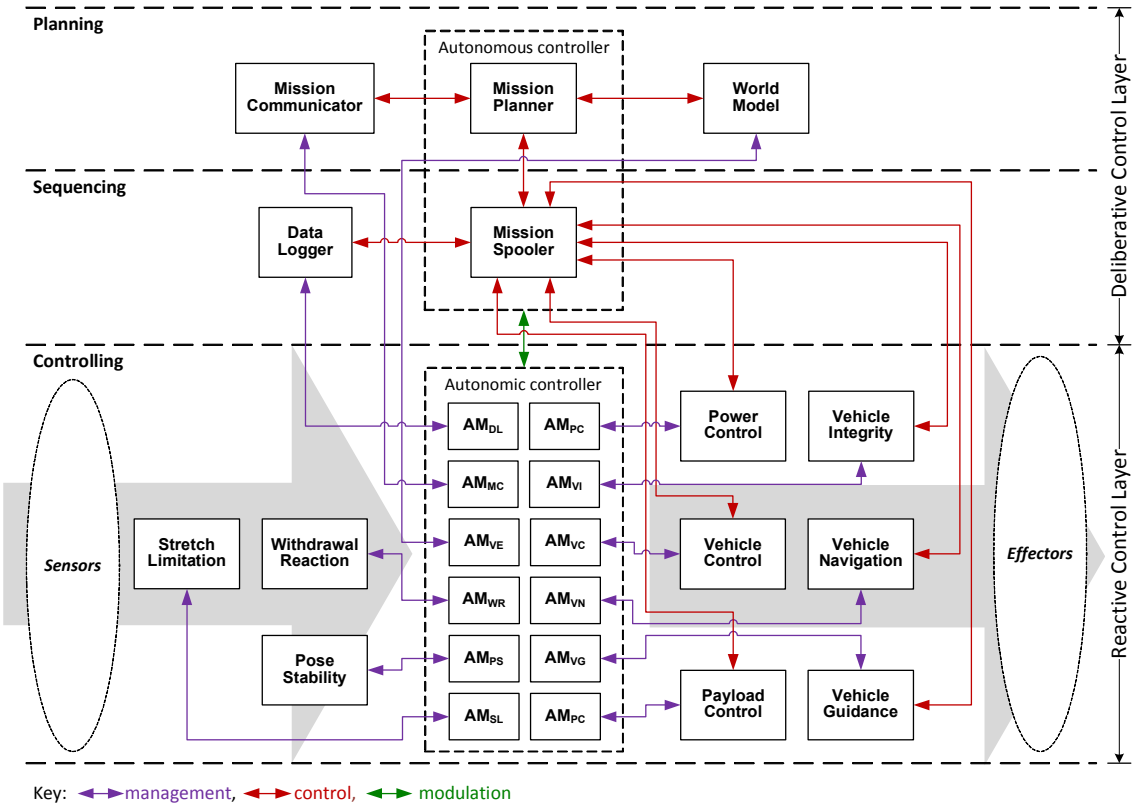

Fig. 5. Autonomic AUV control architecture.

Figure 5 presents a hierarchical control structure where an autonomous controller (supervisor; central control, i.e. the mission planner and the mission spooler) has more authority than an autonomic controller (manager; autonomic management, i.e. set of Autonomic Managers: MA_{Module}). This is not a master-slave control paradigm but it is rather the former modulating the latter's controlling capabilities. The MEs are the classical AUV modules plus the following modules: stretch limitation, withdraw reaction, and pose stability. The classical MEs plus their AMs deal with the internal AUV reflexes, and the remainders plus their AMs deal with the external AUV reflexes.

The AM has a direct connection with the vital system modules (organic analogy). It looks after the modules such as navigator, pilot, etc. by keeping the overall system updated as to the module statuses, performance, configuration, and vulnerability. The AM receives information from the modules and also from other equipment and devices of the system. There is an automatic reaction for eventual occurrences on the modules. They are quick responses to critical events occurred either in the internal or external environment.

The AMs automatically accomplishes an auto-regulatory control in the system by managing the modules. This self-regulation is well supported by the four self-managing capabilities (i.e. self-healing, self-protecting, self-optimizing, and self-configuring) as explained in the next Subsection.

The AM can be integrated to other architectural configurations, including standard architectures such as the Joint Architecture for Unmanned Systems (JAUS) [19], an open architectural approach for the domain of unmanned systems, and the Reference Model Architecture (4D/RCS) for military unmanned vehicles on how software components can be identified and organized [20].

B. Self-managing Capabilities

1) Self-healing Capability

Self-healing is the ability to diagnose and mitigate system malfunctions so that reliability and consequently availability can be guaranteed. This capability makes it possible to observe failures, evaluate internal and/or external constraints, and apply appropriate corrections. Discovering malfunctions requires knowing the expected system behaviour. Self-healing systems endure operations of missions in order to maintain satisfactory quality of service during the system runtime while faults are present [18].

The self-healing process within the MAPE control loop of the AC paradigm is as follows. The AM monitors the system behaviour according to the data received from the sensors. Any improper system behaviour is detected according to the knowledge and policy. If no fault is detected, the MAPE loop dies in the monitoring stage. Otherwise, the error is sent to the analysis module to diagnose the symptom. The diagnosis result is sent to the plan module in order to specify a repair operation to be executed. Once selected the mitigation operation, it is executed by the execution module though the effectors. The loop is closed by continuously sensing the status of the system components [3].

2) Self-optimizing Capability

Self-optimizing means tune-up and continuous monitoring of system capabilities so that resource allocations and workloads can meet requirements as well as efficiency and effectiveness. This capability provides self-optimization as to energy consumption, mission performance, and resource allocation.

The energy utilization involves any energy (electrical, mechanical, etc.) used and transformed. The good use of the energy extends the battery duration so efficient energy consumption becomes critical for persistent presence in missions. Therefore, optimizing the energy use means longer missions.

The time to carry out a mission counts in optimization but there is a trade-off between the time to the mission and the quality of service, e.g. seabed data quality vs. vehicle velocity. The Centre of Gravity (CoG) of the vehicle influences on its hydrodynamics which in turn can affect the optimum point to efficiently go through the water (pose stability). In addition, the vehicle shape counts when the moving resistance is taken into account for optimization.

The footprint involves optimal allocation of processing hardware and computing software as needed.

The implementation of self-optimization can be done by means of situation-action rules for all situations that can occur, goal policies that divide the states of the system into desirable and undesirable ones, and utility functions.

3) Self-configuring Capability

Self-configuring means automatic reflection of, and adaptation to the dynamically changing environments.

If the water is not clear in a seabed survey, the AUV should be able to automatically switch from a visual servoing to an acoustic one. Also, self-configuration involves capability of autofocus of the on-board cameras in order to capture better images. The change on the vehicle shape (fin, docking other vehicles), position of the thrusters, adjustment of the cameras position, are included in this capability as well.

Assignment of workload on different resources, e.g. distribution of software processes according to their computational demand. This also includes hardware processes such as manipulation techniques.

4) Self-protecting Capability

Self-protecting means anticipation, detection, and identification from internal and external attacks so that security can be assured at any time.

The typical protection is against a virus or a hacker of computer. The system integrity is a key factor in term of security issues. Protection does not only mean to prevent the system from intrusion, it also means to be prepared for disruption of, for example, the communication channel (interference caused by strange agents) that prevents contact with the rest of the system.

Protection also involves collision avoidance (withdrawal reaction) as well as any risky manoeuvre (stretch limitation) in order to prevent the vehicle from any physical damage. Protection of the vehicle from leaking (vehicle integrity) as well as overheating is part of the self-protecting capability.

C. Autonomic Control Paradigms

The control algorithms analysed in this Subsection address different strategic aspects such as adaptation, optimization, hierarchy, prediction, robustness, and intelligence.

Table II shows how the above control strategies are mapped on the self-* capabilities, i.e. what kind of control algorithm is suitable to deal with, and implement such capabilities.

TABLE II. CONTROL STRATEGY REQUIRED ACCORDING TO AC PROPERTIES

Control strategy	Self-* Capability				Derived Self-* Capability			Expanded Self-* Capability	
	Configured	Healed	Optimized	Protected	Adjusted	Situated	Aware	Anticipatory	Open
Adaptive	X					X		X	
Optimal			X		X				
Robust		X		X					
Hierarchical									X
Stochastic			X					X	

D. Autonomic Software Platform

There are several software platforms as potential candidates for the implementation and integration of an AM with a central controller of autonomous systems. Software platforms involve software development tools, and middleware (including operating systems). What it is needed is support to implement different communication models, well support for multi-processing, and low latency when transmitting-receiving data. These are required to implement the fast-response reflex mechanisms of the MAs.

Development tools such as Open Robot Control Software (OROCOS) [24] and Coupled-Layer Architecture for Robotic Autonomy (CLARITy) [25] meet the above requirements. Robotic middleware such as the Robot Operating System (ROS) [26], and general purpose publish-subscribe middleware like Data Distribution Service (DDS) meet the above requirements as well. They are not the only ones that can deal with the AM requirements but few to be mentioned.

V. AUTONOMIC MANAGEMENT

This Section describes the AM functionality in the AUV control architecture. The description is focussed on some application examples of implementation of self-* capabilities.

A. Autonomic Healing

The implementation of the self-healing is done by defining and specifying possible faults, its diagnosis, and a mitigation plan at the two control layers. Thus, a repairing at the reactive and deliberative layers is carried out under the management of the corresponding MAs. Two different faults are considered: malfunctions in AUV software and/or hardware (vehicle), and disturbances due to environmental conditions (environment).

Table III presents some faults that could be handled through the self-healing capability. This table shows some examples of type of faults that can arise at the deliberative control layer and from (1) vehicle problems or (2) environmental conditions; or at the reactive control layer, and from (3) vehicle problems or (4) environmental conditions.

TABLE III. EXAMPLES OF FAULTS DIAGNOSED AND MITIGATED AT THE CONTROL LAYERS

Cause	Diagnosis	Mitigation	Policy	Type
Marine Current	Wrong path or pose	Pose and motion compensation	If the AUV path or pose is incorrect, the AM actives a compensation mechanism that takes into account the path or pose given as reference	(4)
Mechanical Problem	Module is not responding	Mechanical/Electronic redundancy, alternative capability	If a piece of mechanical hardware or electronic module fails, the AM evaluates the situation in order to activate a redundant system or an alternative capability to replace the faulty one	(3)
Electronic Problem	Device is not working			
Acoustic Interference	Bad acoustic communication or image	Quality analysis and improvement	If there is an acoustic disturbance, the AM analyses the situation in order to active filter and prediction mechanisms to improve acoustics. If this does not work, it assists the AUV planner to make a decision on what to do next	(2)
Evitable Collision	Obstacle detection	Collision-free operation mode	If an obstacle is detected as a potential threat for collision, the AM actives a collision avoidance mechanism that can or cannot involve the AUV planner	(2/4)
Inevitable Collision	Collision with, e.g. vessel, seafloor, other AUV	Collision-recovery operation mode	If the AUV unavoidably collides with something, the AM actives a collision recovery mechanism that takes first the AUV away, and then starts a built-in integrity test	(4)
Operational Problem	Identification, classification, and analysis of the fault	Recovery mechanism	Depending on at what hierarchical level the fault occurs, the AM makes the decision on what is next, or if the AM cannot deal with the problem, it can assist the autonomous controller to deal with it.	(1)

B. Autonomic optimization

The implementation of the self-optimization deals with the hydrodynamic efficiency and AUV effectiveness in order to decrease the energy consumption. The AUV shape is designed to provide as little friction with the water as possible. As an implementation example, the AM makes sure that the following optimization processes are automatically carried out:

- Horizontal vehicle motions are done by heading forward with direction to the final destination, e.g. a simple positioning operation has to pose the vehicle according to the vector given by the displacement direction, and then the vehicle can re-pose as needed.
- Straight paths and shortcuts are taken when possible. This also favours to the reduction of the mission duration.
- Vertical and lateral motions, including roll, yaw, and pitch manures are done by adjusting the pose of the vehicle fins so that the water resistance can be reduced.
- The CoG of the vehicle is at an optimum point to efficiently go through the water since it influences on the hydrodynamic features of the vehicle.
- Devices, instruments, and computer as well as sensors that are not used, are switch off during the mission. This also links with an optimal allocation of resources.

The last three items are adaptation mechanisms, and can also be seen as self-configuration capabilities of the AUV.

C. Autonomic Configuration

The implementation of the self-configuration deals with the adaptation of the AUV in order to improve the satisfaction of mission requirements. Automatic reflection of dynamically changing environment requires the AM, among other things, to make sure that the following optimization processes are automatically carried out:

- The autofocus capability is running on the on-board cameras in order to capture better images. This also includes any pose adjustment of the cameras.
- The vehicle lights are switched on if the water is not clear in a visual seabed survey. If there is still problem with the visibility, an automatic switch from the visual servoing to the acoustic one could be suggested by the AUV(s).
- The self-setting of vehicle parameters for a better adaptation to the underwater environment, e.g. density, pressure, and temperature of the water.
- The configuration of the thrusters to lead the vehicle to different waypoints, assuming the vehicle has reduced propulsion capabilities (only for specific directions).
- Self-assignment of workload on different resources, e.g. distribution of software processes according to their computational demand.

D. Autonomic Protection

The implementation of self-protection deals with the internal and external AUV network security in order to block any attack. Developing a sort of immunological mechanism provides the AUV with capabilities to anticipation, detection, and identification of attacks to assure high security at any time. For example, it requires the AM to make sure that the following protection processes are automatically carried out:

- The system integrity is guaranteed at any time.
- The protection against viruses or hackers of computer.
- Protection does not only mean to prevent the system from intrusion, it also means to be prepared for disruption of, for example, the communication channel (interference caused by strange agents) that prevents contact with the rest of the system.
- Protection also involves collision avoidance as well as any risky manoeuvre in order to prevent the vehicle from any physical damage.
- Protection of the vehicle from leaking as well as overheating is also part of the self-protecting capability.

The above mechanisms are just few integrated solution examples (not limited to) of self-* capabilities required to support operational resilience and efficient energy consumption in AUV(s).

VI. DISCUSSION

AC brings many benefits to IT systems by endowing them with self-managing capabilities. The benefits are at low level (short term) such as a faster IT maintenance response, and a better use of IT resources as well as at high level (long term) such as collaborative global-local solution of IT problems, and massive autonomic regulation of IT infrastructures.

The AC-solution for AUVs has similar expected benefits that the IT ones. The different domains IT systems and AUVs deal with, makes the benefits focus change. The benefits for AUVs are focussed on increasing endurance for missions on the basis the non-stop maintenance and adaptation are supported.

Separation of managing process from the controlling one in the AUV control architecture proposed favours to the efficiency for allocation of tasks but arises up a challenge of parallel process execution that deserve further investigation.

A questionable fact due to the underwater domain is that why not to be inspired by fish instead of human beings. Inspiration in human body (including fish physiology) is a broader approach that assures more autonomous and autonomic capabilities supported as future AUVs will involve massive integration of advanced capabilities for operational endurance and survival.

As a result of machines being capable of replacing some human functions, autonomous systems are inspired by human beings. Most of them are isolated solutions for specific problems but not integrated solutions for full adaptation to an unknown environment. However, some computational IT paradigms such as AC [4], Organic Computing [28], and the Viable System Model (VSM) [29] have tried to integrate the self-* capabilities inspired by human physiology. The lack of maturity and clarity as well as the different application domains limits their adoption in applications of control systems.

VII. Conclusions

An introduction of the AC paradigm as a potential solution for self-management in control architectures of AUVs has been presented. The AC principle comes from IT, and this research work tried to apply a similar concept to AUVs. The differences between the two domains require some analogies in order to keep the idea of self-management unchanged. This paper proposes an AC-based approach for AUV control architecture in order to increase the chance of success for persistent presence in missions.

VIII. Future Work

Future research is divided into two directions.

- A short-term goal is a research effort to fully develop an approach based on AC capabilities, and a solution involving self-organizing capabilities for several AUVs.
- A reference architecture inspired by the most autonomous and highly-intelligent creature (human begin) is very attractive as a good architectural stereotype (fully-integrated approach as to autonomy and autonomic capabilities) for the next generation of any autonomous systems, including AUVs. Therefore, a long-term goal is an effort to investigate and develop a self-managed AUV control architecture more structurally similar to the human physiology.

References

[1] A. Waugh, A. Grant, Anatomy and Physiology in Health and Illness, 9 Ed., Ross and Wilson, 2004.

[2] R. R. Seeley, T. D. Stephens, P. Tate, "Anatomy & Physiology", 8th edition, McGraw Hill, pp. 564-578, 2008.

[3] S. S. Laster, A. O Olatunji, "Autonomic computing: Toward a Self-Healing System", in proc. of the Spring American Society for Engineering Education Conference, 2007.

[4] IBM Corporation, "Autonomic Computing", available in www.research.ibm.com/autonomic, 2001.

[5] P. Horn, "Autonomic Computing: IBM's perspective on the state of information technology", in IBM Research, 2001,

[6] J. R. Boyd, The Essence of winning and losing, 1996.

[7] H. A. Müller, L. O'Brien, M. Klein, B. Wood, "Autonomic Computing", Technical Note, CMU/SEI-2006-TN-006, 2006.

[8] Kephart, Jeffrey O. & Walsh, William E. "An Artificial Intelligence Perspective on Autonomic Computing Policies", in proc. 5th IEEE Int. Workshop on Policies for Distributed Systems and Networks, Yorktown Heights, NY, USA, 2004.

[9] C. Lin, S. Ren, X. Feng, Y. Li, J. Xu, "Autonomic element based architecture for unmanned underwater vehicles", in proc. OCEAN Conf., Sidney, Australia, 2010.

[10] J. Thornton, "Survivability – its importance in the maritime environment", Journal of Defence Science, vol. 10, no. 2, pp. 57-60, 2005.

[11] M. Benjamin, J. Curcio, J. Leonard, P. Newman, "Navigation of unmanned marine vehicles in accordance with the rules of the road", in proc. Int. Conf. on Robotics and Automation, 2006.

[12] W. Truszkowski, H. L. Hallock, C. Rouff, J. Karlin, J. Rash, M. Hinchey, R. Sterritt, "Autonomous and Autonomic Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems", NASA, Springer, 2009.

[13] N. A. Melchior, W. D. Smart, "Autonomic Systems for Mobile Robots", in proc. Int. Conf. on AC, New York, USA, 2004.

[14] E. Vassev, M. Hinchey, "Knowledge Representation and Awareness in Autonomic Service-Component Ensembles – State of the Art", in proc. IEEE Int. Symp. on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, Newport Beach, CA, USA, 2011.

[15] S. Huang, E. Aertbeliën, H. Van Brussel, "An Autonomic Behavior-Based Software Architecture for Mobile Manipulator", in proc. IEEE Int. Conf. on Advanced Robotics and its Social Impacts, Taipei, Taiwan, 2008.

[16] J. Kramer, M. Scheutz, "ADE: A Framework for Robust Complex Robotic Architectures", in proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Beijing, China, 2006.

[17] A. Hernando, R. Sanz, R. Calinescu, "A Model-Based Approach to the Autonomic Management of Mobile Robot Resources", in proc. Int. Conf. on Adaptive and Self-Adaptive Systems and Applications, Lisbon, Portugal, 2010.

[18] Y. S. Dai, M. Hinchey, M. Madhusoodan, "A Prototype Model for Self-Healing and Self-Reproduction in Swarm Robotics System", in proc. IEEE Int. Symp. on Dependable, Autonomic and Secure Computing, Indianapolis, IN, USA, 2006.

[19] JAUS Architecture Framework, Reference Architecture Specification, Vol. II, Part 1, 2007.

[20] 4D/RCS: A Reference Model Architecture for Unmanned Vehicle systems, ver. 2.0, NISTIR 6010, National Institute of Standard and Technology.

[21] T. De Wolf, T. Holvoet, "Towards Autonomic Computing: Agent-Based Modelling, Dynamical Systems Analysis, and Decentralised Control", in proc. Int. Workshop on AC Principles and Architectures, Banff Alberta, Canada, 2003.

[22] S. Smolorz, B. Wagner, "Self-organized Distribution of Tasks inside an Autonomous Mobile Robotic System", in proc. Int. Conf. on Informatics in Control, Automation and Robotics, Noordwijkerhout, the Netherlands, 2011.

[23] J. E. Melvin, "AUV Fault detection using model based observer resuduals", Thesis, Naval Postgraduate School Monterey, California, NPS-ME-98-004, 1998.

[24] H. Bruyninckx, "Open Robot Control Software: the OROCOS project", in proc. IEEE Int. Conf. on Robotics and Automation, Seoul, Korea, 2001.

[25] Coupled-Layer Architecture for Robotic Autonomy (CLARITy), available in https://claraty.jpl.nasa.gov/man/overview.

[26] Robot Operating System (ROS), available in http://www.ros.org.

[27] Data Distribution Service (DDS), available in http://www.rti.com/products/dds.

[28] Organic Computing, available in http://www.organic-computing.org.

[29] S. Beer, Brain of the Firm, 2nd Ed., Wiley, 1994.