# Classes of Control Architectures for AUV: A brief survey

Muhammad Nasiruddin Mahyuddin and Mohd Rizal Arshad

*USM Robotics Research Group (URRG), School of Electrical and Electronic Engineering*
*Universiti Sains Malaysia*
*14300 Penang, Malaysia*
*URL: http://ee.eng.usm.my/urrg*

## Abstract

*This paper provides a brief survey on the control architectures used in the underwater system and robotics research for AUV application. Advantages and disadvantages of each scheme are discussed briefly in the context of performance evaluation, design flexibility and extensibility, sensor or command fusion and integration, robustness and effectiveness of task modules. There are many schemes available and are usually categorised into classes: hierarchical, heterarchical, behaviour-based and hybrid architectures. A new control architecture for AUV is proposed in this paper. The brief survey and the proposed control architecture is a subset of investigation on the effective control architecture for Autonomous Underwater Vehicle (AUV) system to be implemented in USM's Robotic Research Group (URRG)'s hybrid ROV/AUV project.*

**Keyword: AUV's control architecture, hierarchical, behaviour-based , hybrid architectures**

## 1 Introduction

Developments in Autonomous Underwater Vehicle (AUV) have been of great interest to many researchers, engineers and scientist who keen to send an unsupervised vehicle or robot for underwater exploration. Environment surveillance, sea-bottom exploration and oil pipeline inspection are the usual risky applications carried out by AUVs. AUV systems are endowed with different types of sensors, each feeding with different sets of information valuable for decision making. The control architecture for an AUV should be able to perform seamless integration of a wide range of sensors, accurately gauge and monitor the status of the vehicle, perform the stated mission, and preserve itself at all times. This paper presents a survey on various types of control architectures used on AUV, evaluating the advantages and disadvantages of each one of them in the context of ease of performance evaluation, design flexibility and extensibility, sensor/command integration and fusion, robustness, effectiveness of tasks modules and intensity of computational overhead in control systems.

## 2 Classes of Control Architectures used on AUVs

There are various classes or families of control architectures incorporated in the AUV's control system. Some are proven to be successful in certain aspects and others are regarded to be not suitable for mission requirements. From the authors' viewpoint and analysis, there are in fact four classes of control architectures used in AUV's control system:-

- **hierarchical:** contains deliberative elements which emphasize on reasoning and making predictions about the environment from the constructed coherent world model.
- **heterarchical:** contains reactive elements which is based on sense-react principle.
- **behaviour-based:** contains some states to map and think but it will only look ahead when it is acting. It features a combination of reactive elements and state or some deliberative elements. Example of such scheme is subsumption architecture.
- **hybrid:** combines all the three classes of architecture and give birth to a new control architecture that has a symbiosis of deliberative, reactive and behaviour-based elements. It definitely owns states planner and looks ahead in response to react in a long and short time scales. Examples of this class of control architecture are Distributed Architecture for Mobile Navigation (DAMN), A Three-Layer Architecture for Navigating Through Intricate Situations (ATLANTIS) and Servo, Subsumption, Symbolic Architecture (SSS).

### 2.1 Hierarchical Architecture

The hierarchical architecture exhibits a serial control structure consists of higher levels and lower levels whereby only direct communication is possible between them when they are adjacent to each other. **Deliberative architecture** assumes the structure of a hierarchical architecture whereby the planners are arranged into three homogeneous layer; planning, control and diagnostics [1]. The higher levels are responsible for the overall mission goals whilst the lower levels are responsible for problem specific mission [2]. Figure 1

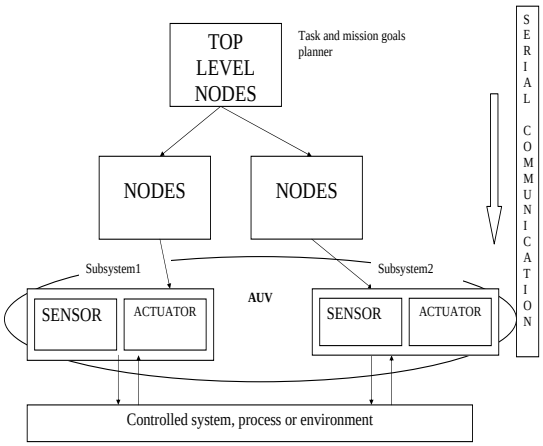


Figure 1: General form of Hiearchical Structure

shows the general structure of a Hierarchical Control Structure.

The advantage of this scheme is that the controllability and stability are easier to be verified which means performance evaluation of the whole architecture can be carried out.

The disadvantage of this scheme in other hand, is that it is not flexible i.e. functionality modifications on certain control levels will lead to significant modifications of the whole system as they are interconnected in serial way. Knowing that the information flow decreases from the bottom to the top of the hierarchy, response time is long and sensor fusion is difficult.

## 2.2 Heterarchical Architecture

Heterarchical architecture uses a parallel structure whereby communication is established between themselves without direct supervision. This architecture is also known as **reactive architecture**.

The advantage of this scheme, as opposed to the hierarchical structure includes flexibility and low communication overhead. Parallel processing is implementable with this scheme since knowledge processing and sensory information can be accessed from any system component.

The disadvantage of this scheme is that the communication among modules can be very intensive and might rise to uncontrollability issue as higher supervision module is absent.

The scheme also has yet to be used in any of the existing AUV research projects. The characteristic of a purely heterarchical architecture makes it unsuitable for AUV use. The Omni-Directional Intelligent Navigator (ODIN) [3] implements a mixed heterarchical and hierarchical architecture for navigational and control.

Figure 2 shows the difference between deliberative(hierarchical) and reactive(heterarchical) architecture in general block diagram.

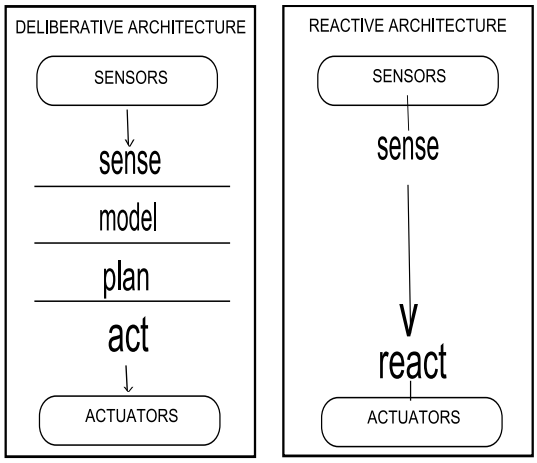


Figure 2: The difference between deliberative and reactive architecture

## 2.3 Subsumption Architecture

Subsumption architecture, introduced by Rodney Brooks [4], is a reactive robot architecture primarily associated with behaviour-based robotics. This scheme decomposes complex and multiple intelligent behaviour into many simple behaviour modules in a layer form, working in parallel. There is no higher level supervision in subsumption architecture. These layers which contain defined behaviour are triggered by sensors in performing an action. One layer can subsume another layer.

The advantage is that lower layer still can function when higher layer fails to function. In terms of extensibility, each layer can have its own processing module as it is decentralised. A layer above may examine the data of the layer below it and may eventually interfere it. This architecture exhibits true dynamic reactive behaviour.

The disadvantage of this scheme is that due to absence of higher level centralized control, system with large number of behaviours becomes complex. The complexity of such scheme associates with synchronization difficulty and timing between behaviours. In contrast with hierarchical architecture, the stability of the system with subsumption architecture is difficult to be verified as most of the modules are decoupled or decentralised. Due to these factors, the implementable version of subsumption architecture is usually modified to give added high-level control functionality which is realised in a form of state configured layered control [5] or formalization [6]. Figure 3 shows the Subsumption Architecture originated and popularised by Brooks.

# 3 Hybrid Architecture

This particular control architecture combines all the three schemes mentioned previously, hierarchical, heterarchical and subsumption architectures. The system comprises of two levels with higher level uses the hierarchical architecture for strategic implementation whilst the lower level uses heterarchical or subsumption architecture to control the hard-

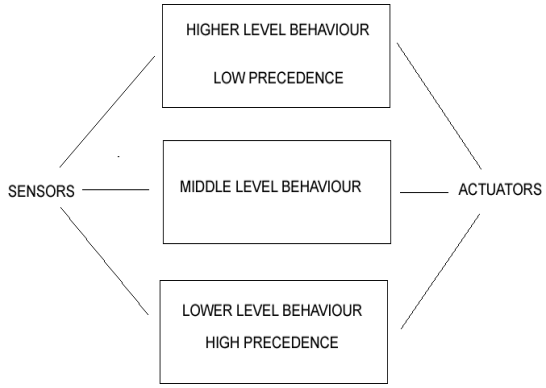


Figure 3: Subsumption Architecture originated by Brooks

ware subsystems [7]. In the lower level as in the case where subsumption architecture resides, the commands from higher levels are translated into series of behaviours which will then be activated. As in the case of heterarchical architecture, several modules performing in parallel fashion and the higher level control above them may interfer or take charge in the case of emergency [3].

The advantage of this scheme is that whilst having flexibility featured by heterarchical (parallel processing modules) or subsumptious architectures (behaviour based layered controls which subsumes tasks) within the lower level, the higher level of the hybrid architecture features greater controllability as most of the lower level multiple operation are synchronised in terms of its communication. Recall from the disadvantage of hierarchical scheme discussion, the communication overhead between adjacent control levels is inevitable in a complex system such as AUV. In this scheme, however, the lower level complements the higher level scheme's demerits.

The disadvantage of this scheme is that the controllability and stability verification are still not easily achieved.

## 3.1 DAMN: Distributed Architecture for Mobile Navigation

This architecture is originally proposed by Rosenbalt [8]. DAMN or Distributed Architecture for Mobile Navigation is a behaviour-based architecture which features some similarities as in Subsumption Architecture. It comprises of multiple modules concurrently share control of the robot by sending votes to be combined rather than commands to be selected and executed. This is different from the hierarchical architecture that employs a top-down structure imposing deliberative elements. According to Rosenbalt [8], the attempt to have a desired symbiosis of deliberative and reactive elements brings about the birth of behaviour based distributed task-achieving modules that cooperate in a bottom-up fashion rather than top-down. Deliberative and reactive elements are fused together so that each complements the other and compensates for the other's deficiencies. Reactive components provide the basic capabilities in carrying out low-level task particularly in responding to unstructured

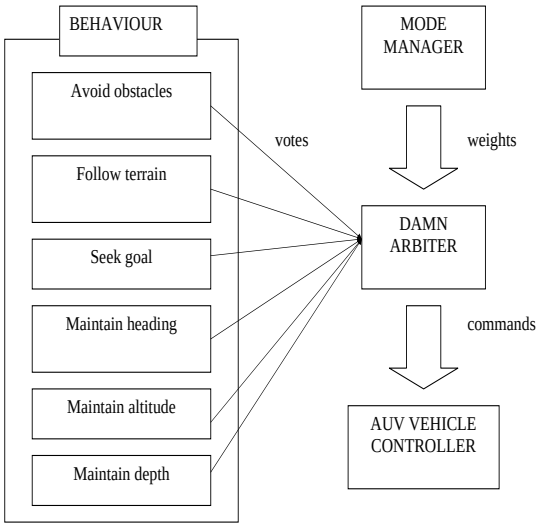


Figure 4: Structure of DAMN

environment, while the deliberative components provide the ability to achieve higher-level goals, associating with mission planning and goals. Figure 4 shows the structure of DAMN.

The advantage of this scheme is that it offers greater reactivity, flexibility and robustness. Being different from purely reactive system, DAMN effectively maintains internal representations of the world while trying to keep the perception and planning components of a behaviour as simple as possible. This scheme replaces sensor fusion with command fusion.

Centralized hierarchical architectures perform sensor fusion in order to construct a coherent world model which then used for planning actions (deliberative elements). Recalling back the features of hierarchical architecture, being able to combine data from previous world models to prevent ambiguities will also create a computationally expensive sensory bottleneck.

By contrast, perceptual processing is employed across the distributed independent modules in DAMN. Each behaviour requires only fragmentary knowledge of the world. There is no need to fuse all available data into single coherent world model. Each behaviour produces a desired action and outputs are produced independently from these actions to control the robot.

## 3.2 ATLANTIS: A Three-Layer Architecture for Navigating Through Intricate Situations

This scheme takes similar structure as that of DAMN where a hybrid deliberative/reactive robot architecture is employed. Some term it as a behaviour-based control architecture. It is a much older behaviour-based architecture than DAMN. The architecture is utilized by Gat [9] to control real-world and simulated real-world robots. It is used to pursue multiple goals in real time in a noisy, partially unpredictable environment. To date, ATLANTIS has not yet been tested on any

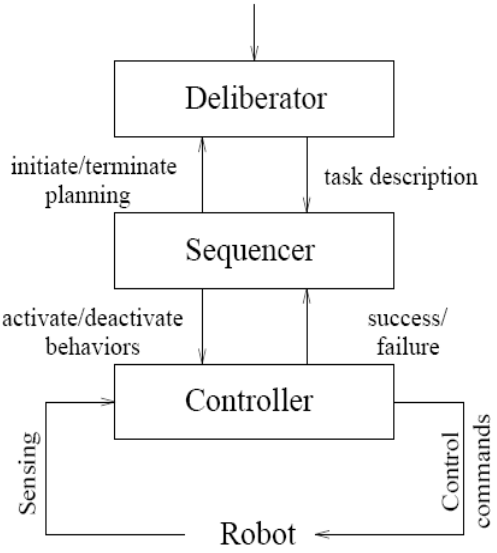


Figure 5: ATLANTIS architecture

of developed AUV in the world. The purpose of this paper to review this architecture is to give a brief insight on the possible application of this scheme on AUV. Based on Gat's conclusion [9], there are several inferences which can be also applied on an AUV system. They are:

- **heterogenous structure:** Using different computational mechanisms to perform different tasks.
- **asynchronous:** A continuous rather than a discrete action model should be used to allow actions to overlap or to be terminated before completing in response to unexpected situations.
- **Abstraction:** A powerful tool for dealing with unpredictable aspects of environment in contrast to classical techniques which involve centralized world model construction.
- **Plans should be used for guide:** Plans should not fully direct and dictate the control and action.
- **Robot control systems should be designed bottom-up:** This particular notion was verified by Rosenbalt in his paper [8].

Figure 5 shows the structure of ATLANTIS architecture.

### 3.3 SSS: Servo, Subsumption, Symbolic Architecture

SSS architecture proposed by [10] closely resembles to the ATLANTIS system. This architecture still adopts a subsumption-style 'control' layer, an operating system-like 'sequencing', and a model-building 'deliberative' layer. It combines three control techniques which can be characterised by their treatment of time and space. As shown in Figure 6, the three layers come from progressively quantizing space at first and then time. According to Connell [10], the

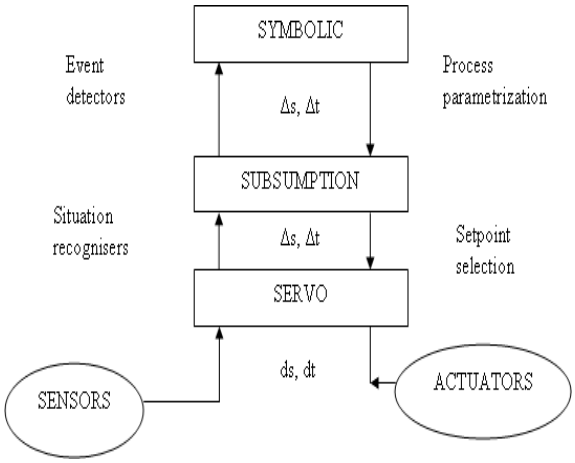


Figure 6: SSS: servo, subsumption, symbolic architecture

difference between ATLANTIS and SSS architecture is that ATLANTIS emphasizes on the sequencer layer into which a partially ordered 'universal plan' is downloaded from deliberative layer. The symbolic system is completely out of the control loop during the actual performance of the prescribed task. SSS in the other hand decouples the symbolic system from the most rapid form of the decision making and thus, still constantly replan the strategy and monitor the execution of each step. Both architectures, SSS and ATLANTIS have not been tested on AUV.

## 4 Proposed Control Architecture based on the survey

Table 1 shows the criteria matrix formulated based on the brief survey carried out.

| CONTROL ARCHITECTURE | | | | |
|---|---|---|---|---|
| | **Hierarchical** | **Heterarchical** | **Subsumption** | **Hybrid** |
| 1 | structured | unstructured | both | depending on the world map built |
| 2 | thinking time | reflex | mix | mix |
| 3 | complete verification | hard | very hard | very hard |
| 4 | high | high | medium | command fusion is emphasized |
| 5 | high | none | low | level priority dependent |
| 6 | not extensible | modular | modular | modular |

Table 1: Control Architecture Selection

| AUV's Application Characteristics | |
|---|---|
| 1 | Environment Suitability |
| 2 | Run-Time Constraints |
| 3 | Correctness and Completeness |
| 4 | Sensor Fusion Complexity |
| 5 | Computational Overheads |
| 6 | Task Modularity and Algorithm Extensibility |

Table 2: Legend for Table 1

All classes of control architectures are evaluated firstly,

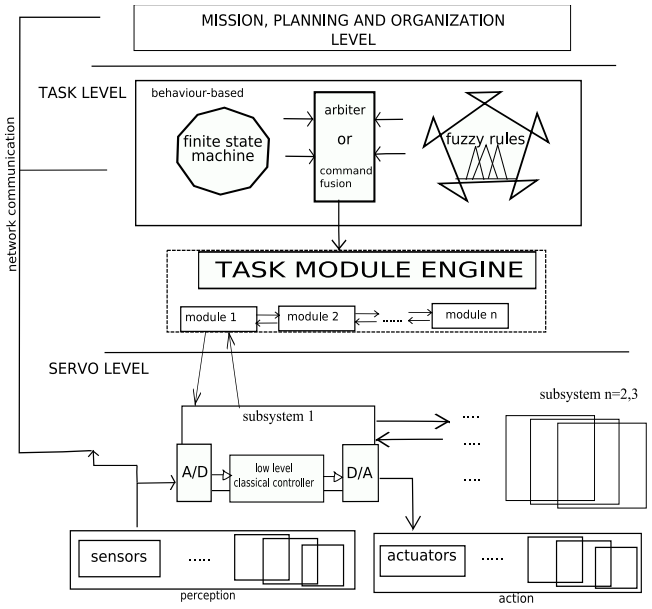


Figure 7: Proposed Control Architecture for URRG's AUV

based on the suitability of each of them to be applied on either structured or unstructured environment, the run-time constraints i.e. overprocessing of information hinders the demand for on-time action/reaction, the verification for correctness, issue of sensor fusions, computational overheads i.e. this relates to the run-time constraints and lastly how modular can we built the task for specific job so that changes on one task module will not affect the overall functionality of the whole control structure.

Figure 7 shows the newly control architecture proposed for URRG's AUV system. The architecture is of hybrid form, combining hierarchical type for the higher level(reserved for mission planner) and heterarchical with distributed behaviour based architecture which resides at the middle and lower level.

The task level comprises of behaviour-based higher control layers consisting several behaviours defining symbolic action. The unique characteristic of this layer is that the behaviours are defined in two knowledge-based units:-

- Finite State Machine
- Fuzzy Rules Engine

Finite State Machine contains libraries of behaviours-defining action ready to be invoked whenever fast action (specifically a reaction) is in demand. This serves almost like a look-up table. Fuzzy rules engine in the other hand, fuzzifies the higher level input and determine the weights to be fed to the arbiter so that behaviour modules can be chosen when votes are issued. The concept of voting (represented by Figure 4) issued by each of the behaviour modules originated from DAMN [8]. However, Rosenbalt leaves an ample amount of room for other researchers to try out different methods of optimizing the weights issued by the mode manager which is to be fed to the arbiter. Therefore, in this paper, fuzzy rule engine is proposed as the weight adjustment mechanism to work together with the DAMN arbiter.

The command fusion will be also influenced by the fuzzy rule engine as to either simply switch to finite state machine (for less computation effort, less power consumption and presumably faster reaction time) or opt for decision made by the fuzzy-rule engine for more demanding mission requirement. This translates to better option for run-time performance.

The task module engine, which is connected to series of subsystem interacting with the actuators and sensors, is made to be modular and distributed. Each of the module can be independently designed by different designers/researchers and once the module is finished, it can be easily integrated into the main AUV's control system without affecting both the lateral and vertical control structure integrity. This helps to accelerate the development process of an AUV project. Although inter-communication exist between adjacent modules, modularity can be preserved as they are structured to be in heterarchical manner, thereby processing data in a parallel fashion. Communication overhead can be eliminated as much of the data processing and flow are carried out in bottom-up manner. The communication traffic between the task modules can be regulated by the mission, planning and organization (MPO) level in a minor circumstance whenever certain parameters meet the threshold. As discussed in the survey (see section 3.3), the higher level i.e. (MPO) level provides guidance but not to dictate the lower level control modules.

Sensors are connected to servo level as well as the task level and the MPO level. Sensor fusion is less of a concern as command fusion accomplished by DAMN-like arbiter is implemented in the task level of the proposed control architecture.

# 5 Summary and Conclusion

Classes of control architectures have been reviewed. There are many more control architectures in the literature particularly in the field of underwater system research that are not mentioned in this paper. Although they are diversed and many, mostly can be categorised into these four classes (heterarchical is not feasible): hierarchical, subsumption, behaviour-based and hybrid control architecture.

A new control architecture for AUV's control system is proposed in this paper. The implementation aspect of the proposed architecture will be preceeded by further analysis work on the mission requirement imposed on the URRG's AUV. The state of the sea of Malaysia in which the developed AUV is to be tested will also affect the mission requirement. An early decision on the most effective type of control architectures to be employed on any AUV system can save the expenses, time and effort of researchers in an AUV project.

# References

[1] P.Ridao, J. Yuh, J. Battle, and K. Sugihara, "On auv control architecture," in *Proc. of the International Con-*

*ference on Intelligent Robots and Systems (IROS 2000), vol.2*, pp. 855–860, 2000.

[2] K. P.Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, and G. A. Demetriou, "Control architectures for autonomous underwater vehicles," *IEEE*, vol. 7, no. 2, pp. 87–105, 2000.

[3] J. Yuh, *Underwater Robotic Vehicles: Design and Control*. TSI Press, Albuquerque, 1995.

[4] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 1, pp. 14–23, 1986.

[5] J. Bellingham and T. Consi, "State configured layered control," in *Proc. of the IARP 1st Workshop on: Mobile Robots for Subsea environments*, pp. 75–80, 1990.

[6] A. Boswell and J. Leaney, "Using the subsumption architecture in an autonomous underwater robot: Expostulations, extensions and experiences," in *Proc. of the IARP 2nd Workshop on: Mobile Robots for Subsea environments*, pp. 95–106, 1994.

[7] A. Healey, D. Marco, R. McGhee, B. Brutzman, R. Christi, and F. Papoulias, "Coordinating the hovering behaviours of the NPS AUV II using onboard sonar servoing," in *Proc. of the IARP 2nd Workshop on: Mobile Robots for Subsea Environments*, pp. 53–62, 1994.

[8] J. K. Rosenbalt, "DAMN: Distributed architecture for mobile navigation," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 339–360, 1997.

[9] E. Gat, "Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots," in *Proc. National Conference on Artificial Intelligence (AAAI)*, pp. 809–815, 1992.

[10] J. H. Connell, "SSS: A hybrid architecture applied to robot navigation," in *Proc. of the 1992 IEEE International Conference on Robotics and Automation, Nice, France*, pp. 2719–2724, 1992.