Design and Implementation of Control Architecture for the ISiMI6000 Autonomous Underwater Vehicle

Banghyun Kim, Pan-Mook Lee, Bong-Huan Jun, Jin-Yeong Park, and Hyungwon Shim
Ocean System Engineering Research Division, MOERI
Korea Institute of Ocean Science & Technology
Daejeon, Republic of Korea
{bhkim, pmlee, bhjeon, jinyeong96, hwshim}@kiost.ac

***Abstract*—This paper presents the design and implementation of control architecture for the ISiMI6000 AUV (Autonomous Underwater Vehicle) which is developing by MOERI-KIOST. The ISiMI6000 AUV is a sea-trial AUV up to the 6,000m depth. The control architecture of the ISiMI6000 AUV is a hybrid architecture consisting of the mission layer, the behavior layer, the logical sensor layer, and the library layer. The mission layer is in charge of the high level control of the AUV using TML (Tiny Mission Language). The TML can represent any mission easily and freely because it provides most functionality supported by general programming language. The behavior layer decides the AUV action by deterministic behavior arbitration mechanism. The logical sensor layer manages input data from sensors, output data from actuators, and environment data for AUV control using shared data pool. The library layer contains many useful libraries for fundamental functions such as hardware interface, communication and real-time management. The real-time management module provides soft real-time characteristic using software timer without real-time operating system. The control architecture has been implemented in two single board computers and two microcontrollers using C language and its software structure is hierarchy and modular.**

***Index Terms*—ISiMI6000, control architecture, autonomous underwater vehicle, tiny mission language, soft real-time**

I. INTRODUCTION

The Maritime and Ocean Engineering Research Institute (MOERI), a branch of the Korea Institute of Ocean Science & Technology (KIOST), has developed Autonomous Underwater Vehicles (AUVs) named ISiMI AUV, ISiMI100 AUV, and ISiMI6000 AUV. ISiMI comes from the name of a small but strong fish from an old traditional story in Korea. In English, ISiMI is the acronym for Integrated Submergible for Intelligent Mission Implementation.

The ISiMI series are AUV platforms that satisfy the various needs of the experimental tests required for the development of control and navigational architectures or software algorithms for underwater flying vehicles. To carry out a number of tests while avoiding many difficulties in field trials, the first version of ISiMI was designed to cruise in the Ocean Engineering Basin (OEB), a facility in KORDI that simulates the ocean's environment [1]. The ISiMI100 was a sea-trial version of original ISiMI up to the 100m depth [2], and the ISiMI6000 AUV can survey the deep sea up to the 6,000m depth.

AUVs have become a main tool for surveying below the sea in scientific, military and commercial applications because of the significant improvement in their performance. The autonomy is one of the most critical issues in developing AUVs. Various control architectures have been studied to help increase the autonomy of AUVs [3]-[6]. They could be categorized into three groups: deliberative architecture, behavior-based architecture, and hybrid architecture.

Deliberative architectures [7] are based on planning using a world model. They allow reasoning and making predictions concerning the environment. When dealing with a highly dynamic environment, the delay in the response time is the main drawback. Behavioral architectures [8] are also known as reactive architectures. The decomposition is based on the desired behaviors for the vehicle and missions are normally described as a sequence of phases with a set of active behaviors. The behaviors continuously react to the situation sensed by the perception system. As active behaviors are based on the sense-react principle, they are suitable for dynamic environments. However, since each behavior pursues its own goal, reaction actions issued by one behavior may cause another behavior to deviate from its respective goal. As a result, the vehicle behavior is, at times, unpredictable.

Hybrid architectures [4] take advantage of the two previous architectures while minimizing their limitations. They usually consist of three layers: the deliberative layer, the behavior layer, and the control execution layer. The deliberative layer acts over the behavior layer by configuring the particular set of behaviors and the priorities among the behaviors. The deliberative layer determines if the task is being accomplished properly by monitoring sensor information and the output of the behavior-based layer.

This paper presents the design and implementation of control architecture for the ISiMI6000 AUV, which is extension of the control architecture for the ISiMI100 AUV. The control architecture is a hybrid architecture consisting of the mission layer, the behavior layer, the logical sensor layer, and the library layer. The control architecture has been implemented in two single board computers and two microcontrollers using C language and its software structure is

hierarchy and modular. The implementation for our control architecture is named EmSIA (Embedded Software for ISiMI AUVs), and the part for monitoring and remote control is named ReSIA (Remote System for ISiMI AUVs).

The remainder of this paper is organized as follows. Section II introduces our ISiMI6000 AUV, and Section III describes the control architecture of the ISiMI6000 AUV, and the last section provides concluding remarks and the future of our work.

II. Overview of ISiMI6000 AUV

The ISiMI6000 AUV has 4.65 m length and 0.6 m diameter, and 809 kg weight in air. The operating duration of the ISiMI6000 AUV was designed at 4 hours with lithium-polymer batteries of a 40Ah capacity, and the designed maximum speed of ISiMI6000 AUV is about 4 knots.

The ISiMI6000 AUV system is divided into a mechanical system, a control system and a communication system. The mechanical system includes hull structure, thruster and control fins. The ISiMI6000 AUV has a torpedo-type appearance as shown in Fig. 1. To minimize the drag force coefficient, front section and middle section were designed cylindrical type, and head cone and tail section was designed based on the Myring hull profile equations [9], which are known as the best contours for minimizing the drag coefficient for a given ratio of body length and diameter. Linear stepper motors were chosen for the fin actuators. The pair of rudders is controlled by a linear stepper motor, and the two stern elevators are independently driven by two linear stepper motors.

Fig. 1. Ballasting appearance of the ISiMI6000 AUV.

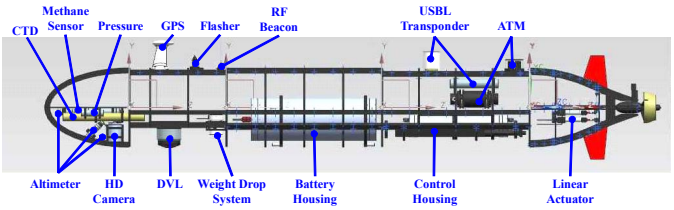

Fig. 2. Instruments configuration.

The control system includes two single board computers (SBC), two microcontrollers, electrical interface boards, and sensors. One computer is for vehicle control, and the other computer is for navigation. Each computer has a PC/104+ type with Intel Atom N450 1.66GHz CPU and its operating system is Microsoft Windows 7. Two microcontrollers were installed to control the digital-to-analog interface, analog-to-digital interface, weight drop system, the linear stepper motors, and thruster. The ISiMI6000 AUV equipped with an Attitude Heading Reference System (AHRS), three altimeters, a Conductivity Temperature Depth (CTD) instrument, a Doppler Velocity Logger (DVL), a HD camera, a Global Positioning System (GPS), an Inertial Measurement Unit (IMU), 7 LED lights, a methane sensor, a pressure, side scan sonar, and an Ultra Short Base Line (USBL) transponder as shown in Fig. 2.

The communication system includes an Ethernet hub, a wireless LAN (WLAN) adapter, a Radio Frequency (RF) modem, and an Acoustic Telemetry Modem (ATM). Wireless devices and RF modem are used for surface communication in air between the ISiMI6000 AUV and a surface unit. The WLAN communication is used for high speed data transfer while the RF communication is used for long distance communication. The ATM is used for underwater communication between the ISiMI6000 AUV and a surface unit.

The surface unit consists of a wireless router, a RF modem, an ATM modem, an Agent computer, and a Remote Control computer. The Agent computer interfaces the ISiMI6000 AUV with the surface unit. All communication devices are connected to the Agent computer. The operator monitors and controls the ISiMI6000 AUV using the Remote Control computer. Fig. 3 shows the relationship of computer and communication system between the ISiMI6000 AUV and the surface unit.

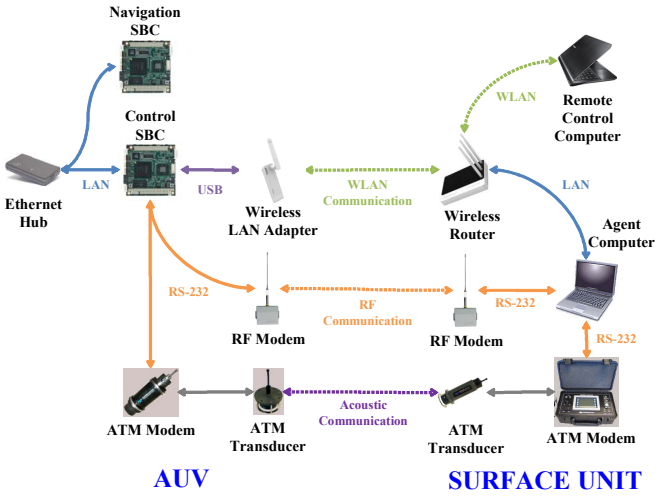

Fig. 3. Computer and communication system.

III. Control Architecture

A great number of control architectures have been implemented, applied to mobile robots, underwater robots, robots for planetary exploration, and so forth. Furthermore, in all these cases, three main approaches are being applied. The deliberative architectures are strongly based on planning and also on a world model, which allows reasoning about and making predictions about the environment. The behavior architectures, also known as reactive, are suitable for dealing with a highly dynamic world difficult to model. The hybrid architectures take advantage of the two previous types, minimizing their limitations. Usually, they are structured in three layers: firstly the deliberative layer, based on planning, secondly the control execution layer (set on-set off behaviors) and thirdly the functional reactive layer [4].

The control architecture of the ISiMI6000 AUV is a hybrid architectures organized into four layers as shown in Fig. 4:

•Mission layer: is in charge of the high level control of the AUV during the mission. It is responsible for the mission planning, execution and supervision. The Tiny Mission Language (TML) [10] has developed for this layer.

•Behavior layer: coordinates active behaviors. The AUV action starts from this layer.

• Logical sensor layer: does not include the hardware interface to sensors and actuators, but manages their data through virtual communication buffer. The data can be physical or logical.

• Library layer: contains many useful libraries for fundamental functions of the AUV such as hardware interface, communication and real-time management.

The mission layer was implemented by the TML virtual machine, which decodes and executes the executable image of a TML mission file. While the mission file is text file, the executable image consists of predefined binary codes. The TML compiler compiles the mission file into the executable image at a surface computer. The executable image is downloaded into the program memory in the virtual machine, which decodes and executes instructions in the program memory sequentially. Some behaviors can be inserted or deleted in the behavior pool by the result of execution.

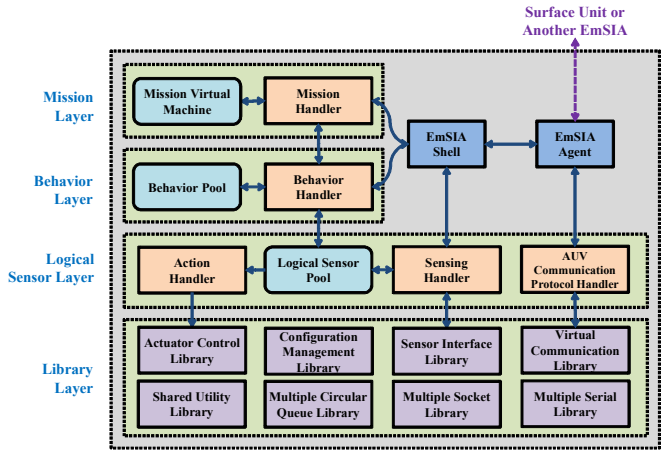

Fig. 4. Control architecture.

The Tiny Mission Language (TML) was designed for the ISiMI AUV series because we need own mission language suited to our control architecture. The main design concept is that the mission language is concise, stable, powerful, easy and fast. Considering these basic design concept, we formulated its design goals as follows:

- Concise: The number of instructions should be less than 64, and the grammar should be simple.
- Stable: The grammar check should be strict, and the minimum syntax check should be possible using a simulator.
- Powerful: The TML should provide control statements for management of execution sequence, variable usage for temporary data and special instructions for behavior management to represent any mission easily and freely.
- Easy: It should be easy to learn the TML and to make a mission file.
- Fast: Decoding and execution time of the TML should be minimized.

Considering these design concepts, we decided that the TML is assembly style because the assembly is most simple in the programming languages and its implementation is easy. The TML also adopted the virtual machine concept to minimize the decoding and execution time. The mission file is converted to the executable image using the TML compiler, and the executable image is loaded to the program memory in the TML virtual machine. Instructions within the loaded image are executed sequentially on the virtual machine as the real machine. A TML mission file for the lawn-mower survey is shown in Fig. 5.

```
# TML example: Lawn-mower type survey
          BHA       BhKeepSpeed, 100     # cm²/s
          BHA       BhKeepAltitude, 100  # cm
          WPA       100, -100, 10        # m
          WPA       100, -100, 10
          WPA       0, -100, 10
          WPA       0, -50, 10
          WPA       100, -50, 10
          WPA       100, 0, 10
          WPA       0, 0, 10
          IN        R1, TIME             # R1=Start time
          BHA       BhWpTrace
L10:      SUS
          IN        R2, TIME
          SUB       R2, R1               # R2=Running time
          CPI       R2, 500              # Timeout=500s
          BRLE      L10
          BHA       BhEmergency
          SUS
```

Fig. 5. Mission example of the lawn-mower survey.

The behavior layer decides the AUV action by a behavior arbitration mechanism [11] based on priority. The behavior handler selects and performs active behaviors having unique priority from high priority to low priority. The action of behavior can change values in logical sensor pool or activate other behaviors having lower priority than current priority. If the reference value related to an actuator is changed, the direction or speed of the AUV may vary.

We defined 14 behaviors for the ISiMI6000 AUV as shown in Fig. 6. The *Emergency* behavior is activated when the emergency instruction of mission language is executed or emergency state is detected. Emergency detection occurs when the altitude is less then defined limit altitude, the location is out of limit range, the remaining battery power is low, or the running time is longer than defined timeout. The *Emergency* behavior inactivates all other active behaviors and activates *Stop* behavior which stops the thruster and drops all weights. Because the ISiMI6000 AUV has positive buoyancy without weights, the AUV rises when the thruster stops. The AUV transfers the received GPS location into the surface unit through the RF communication when the AUV is on surface.

The *Waypoint Trace* behavior traces saved waypoints. If saved waypoints remains, it activates the *Moving* behavior; otherwise it is inactive by itself. If the AUV reaches the target location, next waypoint is popped from saved waypoints then it is overwritten to the target location of the *Moving* behavior. When the purpose of *Waypoint Trace* behavior is complete, all active behaviors are inactive and the *Stop* behavior is activated.

The *Moving* behavior activates the *Keep Depth* behavior and the *Keep Yaw* behavior and it is inactive when the AUV reaches the target location. The *Keep Altitude* behavior activates the *Keep Depth* behavior and sets the reference depth to calculated value by the reference altitude. For example, if the current depth and altitude are 10m and the reference altitude is 5m, the reference depth is set to 15m.

The *Keep Depth* behavior computes the reference angle of stern by a depth control algorithm then it activates the *Keep Stern* behavior. The *Keep Yaw* behavior computes the reference angle of rudder by a yaw control algorithm then it activates the *Keep Rudder* behavior. The *Keep Speed* behavior computes the reference RPM of thruster by a speed control algorithm then it activates the *Keep RPM* behavior.

The *Stop* behavior stops the AUV thruster motor and it finishes behavior arbitration process. The *Keep Stern* behavior moves the stern to the reference angle and the *Keep Rudder* behavior moves the rudder to the reference angle. The *Keep RPM* behavior operates the thruster motor as the reference RPM.

The *Free Dive* behavior and the *Free Rise* behavior are for deep sea trial. These two behaviors drop a weight and move the AUV without thrust. The *Free Dive* behavior drops a weight when the AUV reaches target altitude. The *Free Rise* behavior is inactive when the AUV reaches on surface. The *Drop weight* behavior is active by these two behaviors.

The logical sensor layer manages input data from sensors and output data to actuators using the logical sensor pool. Logical sensor can be created by sensor fusion or abstraction, and actuators are treated as one of logical sensor. For example, computed altitude, predicted location, running time, reference depth, reference heading, and reference RPM of thruster are not physical.

The library layer provides interface of physical sensors and actuators, multiple serial communications, broadcasting socket communications, real-time management, date calculation, bit operation, multiple circular queue, and so on. The real-time library especially provides the software timer at application level using the system clock. The serial communication and socket communication are combined with the virtual communication concept, which manages all networks using unified ID and buffer.

The main process in the control computer repeats following four steps. The first is the input data update process, the second is the TML virtual machine process, the third is the behavior process and the last is the waiting process for synchronization. The interval of a period is 10ms and the system is sleep in the waiting process until next turn time. Fig. 7 shows the main

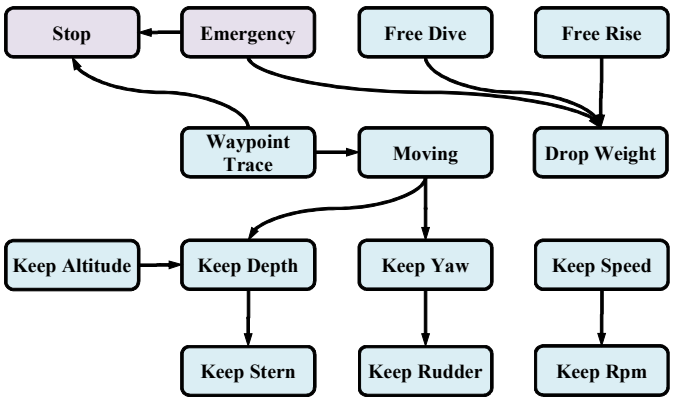

Fig. 6. Behaviors for the ISiMI6000 AUV.

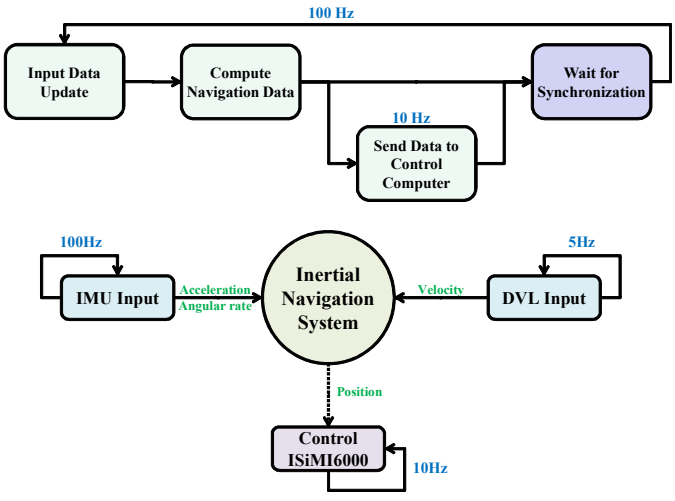

Fig. 7. Main process in the navigation computer.

process in the navigation computer. The basic interval of a period is 100ms for navigation computation. Because all processes should be finished within given period time, the software system of the ISiMI6000 AUV provides a soft real-time module [12] without commercial real-time operating system (RTOS).

Though the ISiMI6000 AUV does not use any RTOS, tasks of control architecture can be synchronized using a user-defined software timer. The timer does not miss desired timing occurrence since the timer was implemented by background thread and system time of SBC. To verify the real-time process using software timer, we performed experiments on Windows 7. The results show that the average error of timing occurrence was almost 0 when the desired control frequency is from 10 Hz to 100 Hz, so our approach using the software timer instead of RTOS is enough to control the ISiMI6000 AUV by desired frequency. Actually, the software timer method is not hard real-time but soft real-time. This approach has no problem because most AUV applications do not need hard real-time characteristics.

The entire software system of the ISiMI6000 AUV consists of the embedded software in the AUV and the remote software in the surface unit. The embedded software can run on either the control computer or the navigation computer according to start parameter. While the embedded software only supports text interface, the remote software provides graphic user interface.

IV. Conclusion

This paper presented the design and implementation of control architecture for the ISiMI6000 AUV, which is extension of the control architecture for the ISiMI100 AUV. The control architecture is a hybrid architecture consisting of the mission layer, the behavior layer, the logical sensor layer, and the library layer. The mission language can represent any mission easily and freely because it provides most functionality supported by general programming language. It provides control statements for management of execution sequence, variable definition for temporary data and instructions for behavior management. The behavior arbitration rule is that active behaviors having unique priority are executed sequentially from high priority to low priority. Another is that running behavior can activate or inactivate other behaviors having lower priority than current priority. This arbitration mechanism based on priority is deterministic and robust. The execution of active behaviors repeats periodically and the execution time of a cycle is less than 10ms to keep control frequency as 10Hz.

To guarantee stable cruise of the ISiMI6000 AUV, we tried to minimize the complexity of control architecture and to reduce ambiguity of AUV actions. In the sea trial in the coast, basic tests of the ISiMI6000 AUV were finished successfully. We plan to conduct the sea trial in the East Sea of Korea up to 1,500m depth from September 2012.

Acknowledgment

The authors would like to thank the support by Ministry of Land, Transportation and Maritime Affair (MLTM) of Korea for the Development of an Advanced Deep-Sea Unmanned Underwater Vehicle, and the support by Korea Institute of Ocean Science & Technology (KIOST) for the performance Demonstration of AUV Technology Transfer and Planning Research of Subsequent Project.

References

[1] B. H. Jun, J. Y. Park, F. Y. Lee, P. M. Lee, C. H. Lee, K. Kim, Y. K. Lim, and J. H. Oh, "Development of the AUV 'ISiMI' and a free running test in an Ocean Engineering Basin," *Ocean Engineering* , Vol. 36, No. 1, pp. 2-14, 2009.

[2] F. Y. Lee, B. H. Jun, P. M. Lee and K. Kim, "Implementation and Test of ISiMI100 AUV for a member of AUVs Fleet," *OCEANS 2008*, pp. 1-6, September 2008.

[3] E. Coste-Maniere, H. H. Wang, and A. Peuch, "Control architectures: What's going on?," in *Proc. US-Portugal Workshop on Undersea Robotics and Intelligent Control*, pp. 54–60, 1995..

[4] P. Ridao, J. Batlle, J. Amat, and G. N. Roberts, "Recent trends in control architectures for autonomous underwater vehicles," *International Journal of Systems Science*, vol. 30, no. 9, pp. 1033–1056, 1999.

[5] K. P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, and G. A. Demetriou, "Control architectures for autonomous underwater vehicles," IEEE Contr. Syst. Mag., vol. 17, no. 6, pp. 48–64, 1997.

[6] P. Ridao, J. Yuh, K. Sugihara, and J. Batlle, "On AUV control architecture," in Proc. Int. Conf. Robots and Systems, Japan, 2000.

[7] A. M. Meystel and J. S. Albus, Intelligent Systems, Architecture, Design and Control. New York: Wiley, 2002.

[8] R. Arkin, Behavior-Based Robotics. Cambridge, MA: MIT Press, 1998.

[9] D. F. Myring, "A Theoretical Study of Body Drag in Subcritical Axisymmetric Flow," Aeronautical Quarterly 27, pp. 186–194, 1976.

[10] B. Kim, B. Jun, H. Sim, F. Lee, and P. Lee, "The Development of Tiny Mission Language for the ISiMI100 Autonomous Underwater Vehicle," OCEANS 2010 Seattle, September 2010.

[11] B. Kim, B. Jun, and P. Lee, "Concise Behavior Arbitration Mechanism for the ISiMI100 AVUsm for the ISiMI100 AUV," UT'11 & SSC'11, April 2011.

[12] B. Kim, B. Jun, J. Park, and P. Lee, "Real-time Process without RTOS for the ISiMI100 Autonomous Underwater Vehicle," OCEANS 2012 Yeosu, May 2012.