Article

Not peer-reviewed version

An OOSEM-Based Design Pattern for Developing AUV Controllers

Cao Duc Sang * , Ngo Van He * , Ngo Van Hien , Nguyen Trong Khuyen

Posted Date: 2 July 2024

doi: 10.20944/preprints202407.0154.v1

Keywords: AUV (Autonomous Underwater Vehicle); OOSEM (Object-Oriented System Engineering Method); MDA (Model-Driven Architecture); real-time UML/SysML; UKF algorithm

Article

An OOSEM-Based Design Pattern for Developing AUV Controllers

Cao Duc Sang [1,*], Ngo Van He [2,*], Ngo Van Hien [2] and Nguyen Trong Khuyen [1]

[1] Control, Automation in Production and Improvement of Technology Institute, Hanoi 100000, Vietnam
[2] School of Mechanical Engineering, Hanoi University of Science and Technology, Hanoi 100000, Vietnam
* Correspondence: sangdctech@gmail.com (C.D.S.); he.ngovan@hust.edu.vn (N.V.H.)

Abstract: The paper introduces a control hybrid model that is designed on the basis of OOSEM (Object-Oriented Systems Engineering Method), MDA (Model-Driven Architecture) concepts, RealTime UML/SysML (Unified Modeling Language/Systems Modeling Language), and an algorithm based on UKF (Unscented Kalman Filter). This hybrid model enables the implementation of control elements for autonomous underwater vehicles (AUVs) and can be adapted to reuse for most standard AUV platforms. To obtain this goal, the dynamic model of AUV is integrated with the following specializations of OOSEM/MDA, in which the analysis model is clarified via the use-case model definition and then combines with HA (hybrid automata) to precisely define the requirements for control. Next, the designed model is specialized via real-time UML/SysML to get the core control blocks, which describe the behaviors and structures of control parts in detail. This design model is then transformed into the model of implementation with the assistance of round-trip engineering to conveniently realize the controller of AUVs. Based on this new model, an AUV controller for low-cost turtle-shaped AUVs is implemented that performs tracking for a planar trajectory with accepted feasibility.

Keywords: AUV (Autonomous Underwater Vehicle); OOSEM (Object-Oriented System Engineering Method); MDA (Model-Driven Architecture); real-time UML/SysML; UKF algorithm

---

1. Introduction

Water vehicles have been strongly studied and used in defense in the last century. Autonomous underwater vehicles (AUVs) are especially being developed for civilian applications, e.g., maritime monitoring and patrol, disaster and tsunami warnings, ocean exploration, etc. [1–4].

Control systems for AUVs always have a challenge because they must be connected to dynamic models of the AUVs in water environments with complex disturbances, e.g., nonlinear waves and sea currents. The control system of an AUV can be implemented as a wrapper that associates discrete and continuous blocks and their interactions, called a HDS (Hybrid Dynamic System), and its control behaviors can be visually modeled by specializing in HA (Hybrid Automata) [5–7]. Traditional control techniques, in combination with soft-computational methods, have often been used for unmanned vehicles or robots [8–15]. This combination has also been used for building up control systems for AUVs; for instance, Table 1 summarizes the assessments on key control techniques used for controller design and implementation of AUVs.

Table 1. Assessments on the main control techniques used for the applications of AUVs.

Used control techniques	Assessment of the performance of AUVs for control applications.
Proportional integral derivative (PID) controller [16–18].	This proved to be well adapted to the AUV when tracking horizontal planar trajectories. However, the controller based on PID control law was only carried out to manipulate the AUV in an environment with less disturbances.
Linear quadratic regulator (LQR) [19,20].	This regulator shows average stability, but it was less dynamic than the PID regulator.
Lyapunov stability [21–23].	It is very effective, especially when controlling the heading of an AUV. Nevertheless, stabilization in the neighboring area of desired waypoints was not strong enough to track a planar trajectory.
Backstepping technique [24,25].	This technique shows a high ability to control the oriented angle of AUV in an environment with high perturbations.
Sliding-mode control [26,27].	This technique did not give excellent performance when it was implemented alone in an AUV controller. The chattering nature of this controller shows its inability to adapt to the dynamics of AUVs. To improve this, this controller could be associated with other techniques, such as NN (neural networks) [30–35] or backstepping [28,29].

The above-assessed points let us choose the backstepping technique combined with the PID regulator, which is named the integral backstepping (IB) method, for performing the continuous behavior evolutions of controllers of UAVs. In addition, development costs and reusability need to be considered when constructing an application. Reusability plays a vital role in the process of developing different applications for AUVs. To achieve this objective, the UML will be standardized to visually analyze and design different system components of a system in the software industrial field. Furthermore, the SysML (System Modeling Language) [37], which is based on UML and standardized by OMG, is utilized to perform different steps (analyzing, designing, verification, and validation) in realizing industrial systems in different domains. A drawback of UML/SysML is the lack of parts for modeling the internal continuous behavior evolution of developed systems. Besides, OOSEM [38–40] was originally based on the MBSE approach, which is standardized by INCOSE [41,42], for modeling requirements, designing, analyzing, validating, and verifying artifacts of developed systems. The OOSEM is a development method at the system level that allows facilitating the combination of systems engineering, OO (object-oriented) software engineering, and applying object-oriented technology in a way that profits from the systems engineering process. OMG also formalized MDA (the Model-Driven Architecture) [43,44] to separate specifications of systematic operations from their details related to the way of using the capabilities of the system platform. Up to date, many industrial applications of RealTime UML and SysML for complicated control systems with the above model-based approaches are shown in [44–52].

Hence, the OOSEM and MDA could be specified in combination with the RealTime UML [53–56] and SysML (denoted as RealTime UML/SysML) to particularly describe the artifacts of the implemented AUV controller. Based on the above assessments, this study emphasizes developing a hybrid model wrapper that combines the dynamic model of AUV specified for the OOSEM with the MDA components consisting of CIM (Computation Independent Model) considered as the analysis model, the PIM (Platform Independent Model) represented as the design model, and the PSM (Platform Specific Model) carried out as the implementation model, followed up by the RealTime

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 July 2024 doi:10.20944/preprints202407.0154.v1

UML-SysML, the algorithm based on UKF and HA (hybrid automata), which allows us to intensively deploy the controller for AUV. On the basis of this hybrid model, an AUV controller for horizontal trajectory tracking dedicated to turtle-shaped AUVs is deployed and tested.

2. Overview of AUV Dynamic Modeling and Control Structure

2.1. Dynamics of Underwater Vehicles for Control

The motional parts of underwater vehicles are specified as surge, sway, heave (position components), and roll, pitch, and yaw (attitude components) by SNAME [57] (see Table 2).

Table 2. Symbols of SNAME for modeling the motions of an underwater vehicle.

DOF	Motions	Forces/Moments	Linear/angular velocities	Position/Euler angles
1	Surge	X	u	x
2	Sway	Y	v	y
3	Heave	Z	w	z
4	Roll	K	p	ϕ
5	Pitch	M	q	θ
6	Yaw	N	r	ψ

Based on the model of the general control structure (navigation, guidance, and control) for an underwater vehicle, the AUV kinematics in the navigation frame and the AUV dynamics in the body frame can be respectively described as the first and second equations in (1) [58].

$$\begin{cases} \dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{v} \\ \boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{D}(\boldsymbol{v}) + \boldsymbol{g}(\boldsymbol{\eta}) = \tau(\boldsymbol{v},\boldsymbol{u}) \end{cases} \tag{1}$$

Here: $\boldsymbol{\eta}_1 = [x, y, z]^T$ denotes the position, $\boldsymbol{\eta}_2 = [\phi, \theta, \psi]^T$ – the orientation, $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^T, \boldsymbol{\eta}_2^T]^T$; $\boldsymbol{v}_1 = [u, v, w]^T$ depicts the linear velocities, $\boldsymbol{v}_2 = [p, q, r]^T$ - angular velocities, $\boldsymbol{\nu} = [\boldsymbol{v}_1^T, \boldsymbol{v}_2^T]^T$; $\boldsymbol{M}, \boldsymbol{D}(\boldsymbol{\nu}), \boldsymbol{C}(\boldsymbol{\nu})$ respectively describe inertia, damping and *Coriolis* matrices; $\boldsymbol{g}(\boldsymbol{\eta})$ represents the buoyancy forces and gravity; $\boldsymbol{\tau}(\boldsymbol{v},\boldsymbol{u})$ – control forces and torques, while $\mathbf{u}$ -the inputs of the controller.

A discrete state-space model is required to deal with the evolution of continuous behaviors of the developed AUV that can estimate the vehicle movements through filters, such as particle filters, extended Kalman filters, etc. [59,60]. The evolution of continuous states of the AUV controller is then developed by (2).

$$\begin{cases} \boldsymbol{x}_k = \boldsymbol{f}_{k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{w}_{k-1} \\ \boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k \end{cases} \tag{2}$$

Where: $\mathbf{x} = \begin{bmatrix} \boldsymbol{\eta} \\ \mathbf{v} \end{bmatrix}$; $\mathbf{x}_k$ denotes the k^{th} state variables; $\mathbf{u}_k$ is control inputs to the system, $\mathbf{y}_k$ is its outputs; $\mathbf{w}_k$, $\mathbf{v}_k$, $\mathbf{h}_k$ denote the supplementary process, measured disturbance and the measurement function, respectively.

2.2. General Architecture of an AUV Controller

An autonomous underwater vehicle often has three sub-systems. The first sub-system (guidance) prepares parameters of the desired trajectory for the AUV controller to reach and follow up on; the second navigation sub-system has a responsibility to predict the actual AUV states; and the control sub-system is mainly responsible for creating the control torques and forces for acting on and conducting the AUV. Each sub-system has its own individual tasks to be performed, yet they must coordinately work on the same purpose, allowing the AUV to perform its missions. Figure 1 brings out a BBD (block definition diagram) formatted in the conventions of SysML that describes the interactions of the sub-systems. As previously mentioned, control systems can be seen as a

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 July 2024 doi:10.20944/preprints202407.0154.v1

combination of continuous and discrete behaviors, which are called HDSs (Hybrid Dynamic Systems). As shown by the above AUV dynamic models (1)–(2) and the three sub-systems, and together with the HDS's features, AUV controllers are thus presumed to be HDSs, of which dynamic models can be implemented by hybrid automata.

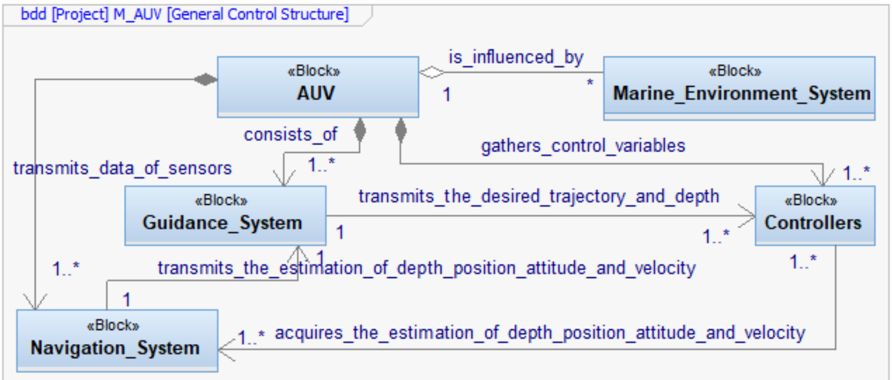

Figure 1. A BDD realized in SysML to represent the above-mentioned three sub-systems of an AUV.

3. OOSEM/MDA-Based Development for an AUV Controller

In this section, the MDA concepts in the OOSEM are specialized for realizing the control parts of an AUV that overall consists of PIM, CIM, and PSM components. All the modeling elements in CIM, PIM, and PSM when realizing AUV controllers will be shown in the following sub-sections.

3.1. CIM realization for AUV Controllers

In CIM, the use case model of RealTime UML-SysML, the functional block diagram, in combination with the PID and IB techniques, and HA are defined to model in detail the requirements of control. Based on the dynamic models and control architecture of AUVs, the key use-case model is defined in Figure 2. The control-oriented modes of an AUV are then modeled by using the state machine extracted from the "Track a desired trajectory" use case. This local state machine is then depicted in Figure 3. Here: MES stands for the Marine Environment System, including various disturbances, for example, waves, wind, ocean current, etc. MDS stands for Measurement Display Systems, which include both guidance and navigation sub-systems.

In the paper, a supplemented FBD (Functional Block Diagram) is defined to provide support to model evolutions of the continuous parts of AUV controllers, as it is hard to utilize the RealTime UML/SysML version, which does not have the standardized notations to visually model complex evolutions of internal continuous behaviors combining with control laws, such as the Integrator Backstepping technique. Thus, a supplemented FBD (Figure 4) was specified for capturing the internal continuous evolution of the control parts of the AUV.

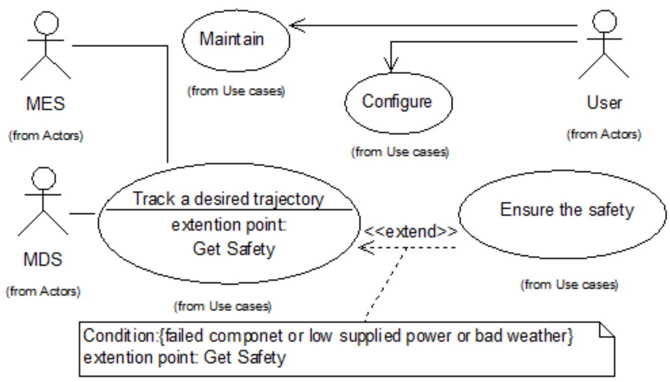

Figure 2. The diagram depicts the use case capturing the core requirements of an AUV controller.

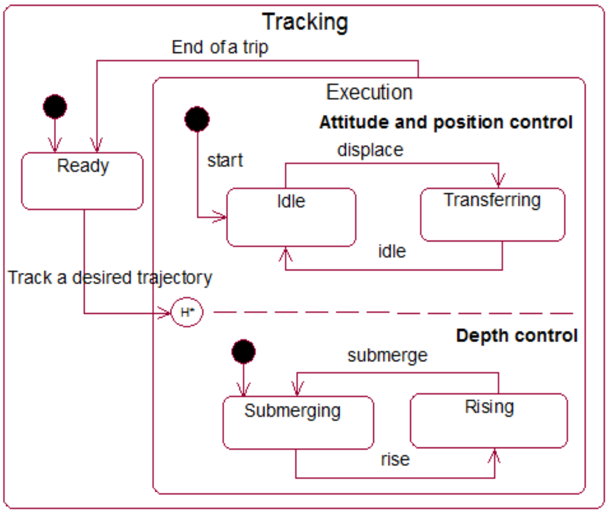

Figure 3. The state machine of the use case "*Track a desired trajectory*".

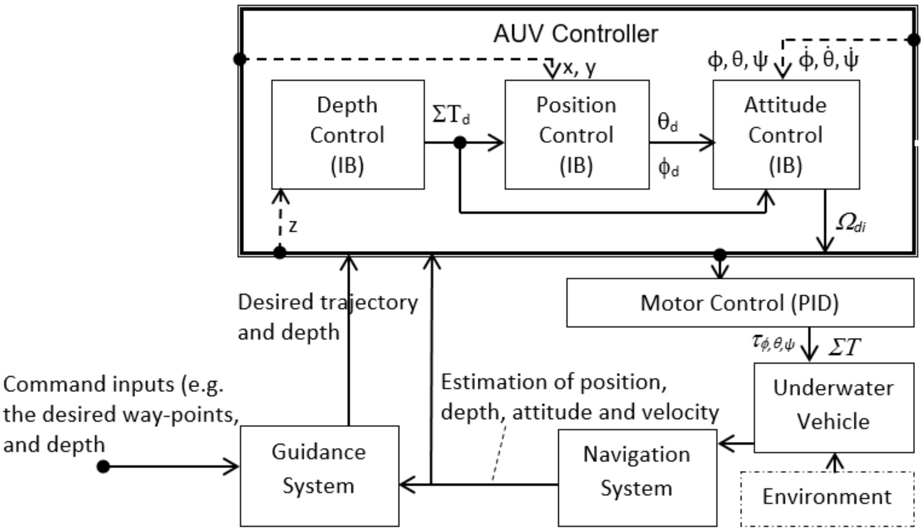

Figure 4. A supplemented FBD for capturing the internal continuous evolution for the control parts of AUV.

Here, *depth* and *desired trajectory* are external events respectively providing the desired position, desired depth to the reference input of the control blocks. Ω_{di}, $i = \overline{1,n}$ are requisite rotation velocity, which are used to conduct motors of the *i* actuator of the AUV, for example, the motors of rudders, propellers and sail plan. $\tau_{\phi,\theta,\Psi}$ and ΣT represent the output torques and forces allocating on the actuators.

In the analysis model as the CIM, HA is applied to globally represent the discrete/continuous models of the AUV controller considered as a HDS that is written as Equation (3).

$$H_{AUV} = (Q, X, \Sigma, A, Inv, F, q_o, x_{co}) \quad (3)$$

Here: Q is the global states (or situations) of AUVs; $q_o \in Q$ - init of Q; X denotes the space of continuous state for describing continuous elements of AUVs, $X \subset \Re^n$; $x_{co} \in X$ - the init of X; Σ - external discrete events, which triggers the transform from the current global state to the target global state. These events are generated from operating environment, external systems, or users, for example, MES, MDS, Users described in the use case diagram (Figure 2); A represent a set of transitions from situations to situations, which are often accompanied by the internal/external discrete events ($\sigma \in \Sigma$); $Inv(q)$ stands for *invariant*, it is an application dedicated to verification if the global state of the system is q. In this case, if the continuous element x_c belongs to the global state q, it must be ensured that x_c

 doi:10.20944/preprints202407.0154.v1

∈ ***inv(q)***; ***F*** are the continuous parts followed up by Equations (1)-(2) that depict the whole continuous evolution to a new situation of the developed AUV. The detailed specializations of HA and evolution hypotheses for the control parts of AUVs can be seen in the report [61].

3.2. PIM Realization for AUV Controllers

As mentioned above, the IB (integrator backstepping) technique, in integration with the CLF (candidate Lyapunov functions), is well performed in various control applications of AUVs. In the PIM, these techniques are used in the Position Control block, Depth Control block, and Attitude Control block (Figure 4), which take part as continuous elements of the AUV controller. The control algorithm based on PID is also applied to the motor control block. However, we will not deal with the details of these control methods because they can be found in a wide range of applications of AUV control [16,17,28,29,62–66].

In addition, the discrete state-space model as described in (2) is used for estimating the states of the AUV. In this study, the state predictions and estimations can be implemented by the standard navigation filter [59], followed by the UKF algorithm as shown in Algorithm 1. Here, $\hat{.}$ represents its estimation; *P* denotes the covariance matrix of the state space; *R*, *Q* represent the measured noise and covariance matrices of the process. The state estimation is performed by the initial conditions: $\hat{\mathbf{x}}_{0|0} = \mathbf{x}_0$ and $P_{0|0} = 0_{12\times12}$.

Algorithm 1. Navigation filter followed up by an UKF filter

Function *UKF-based algorithm*
Step *UKF prediction*
Data : $\hat{\mathbf{x}}_{i-1|i-1}, P_{i-1|i-1}, \mathbf{f}_{i-1}(.)$
Result : $\hat{\mathbf{x}}_{i|i-1}, P_{i|i-1}$

$$\left(\hat{\mathbf{x}}_{i|i-1}, \overline{P}_{i|i-1}\right) = \mathrm{UT}\left(\hat{\mathbf{x}}_{i-1|i-1}, \overline{P}_{i-1|i-1}, \mathbf{f}_{i-1}(.)\right);$$

$$P_{i|i-1} = \overline{P}_{i|i-1} + Q_{i-1};$$

end
Step *UKF updating*
Data : $\hat{\mathbf{x}}_{i|i-1}, P_{i|i-1}, \mathbf{h}_i(.)$
Result : $\hat{\mathbf{x}}_{i|i}, P_{i|i}$

$$\left(\hat{\mathbf{y}}_{i|i-1}, \bar{S}_i, P_i^{xy}\right) = \mathrm{UT}\left(\hat{\mathbf{x}}_{i|i-1}, P_{i|i-1}, \mathbf{h}_{i-1}(.)\right);$$

$$S_i = R_i + \bar{S}_i;$$
$$L_i = P_i^{xy} S_i^{-1};$$
$$\mathbf{e}_i = \mathbf{y}_i - \hat{\mathbf{y}}_{i|i-1};$$
$$\hat{\mathbf{x}}_{i|i} = \hat{\mathbf{x}}_{i|i-1} + L_i \mathbf{e}_i;$$
$$P_{i|i} = P_{i|i-1} - L_i S_i L_i^T;$$

end

Subsequently, the PIM is conventionally designed to generate a pattern of a real-time capsule, which permits visually capturing the control elements in the object collaboration of RealTime UML-SysML. From the CIM components defined above, a set of five capsules of control elements were designed to take part in the HA evolution of the control parts for the developed AUV: the capsule of discrete part, the capsule of continuous part, the capsule of ICGB (Instantaneous Global Continuous Behavior), the capsule of internal interface, and the capsule of external interface. Figures 5 and 6 show the design pattern of various capsules for the AUVs following the conventions of real-time UML-SysML structure and class diagrams.

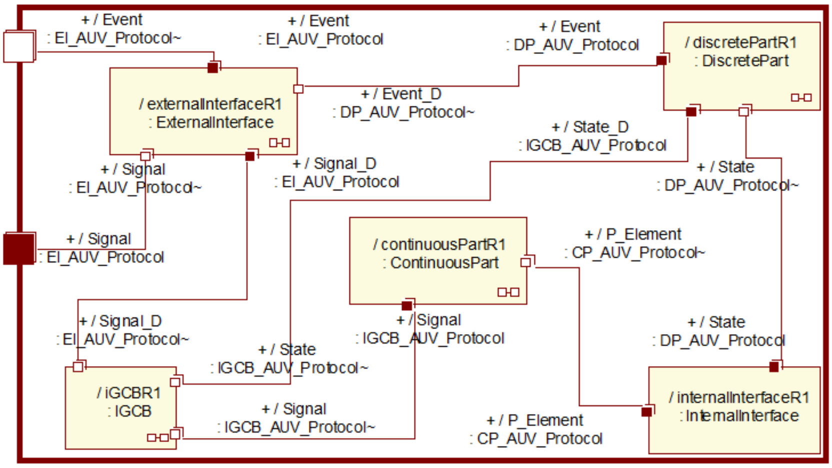

Figure 5. A design pattern depicting control capsules for the developed AUV.

Here: the capsule of discrete part is dedicated to the implementation of transitions A and states Q; the capsule of IGCB is used to form an instant of the continuous models, for example, $f\in F$ is created from the supplemented FBD (Figure 4); the capsule of internal interface uses invariant module (Inv) to generate discrete events appeared in the continuous and discrete evolutions of the HA; the capsule of the continuous component deal with continuous evolution going on in the state space X; the capsule for external interface exchanges continuous signals and discrete events between this designed AUV controller and the external elements, for example, the MDS, MES, or system of User depicted in the use case diagram (Figure 2).

In addition, the capsules of discrete parts and IGCB have their own local state machines, as shown in Figures 7 and 8, to cooperatively synchronize and validate the realization evolution of HA for the five main control capsules in this pattern.

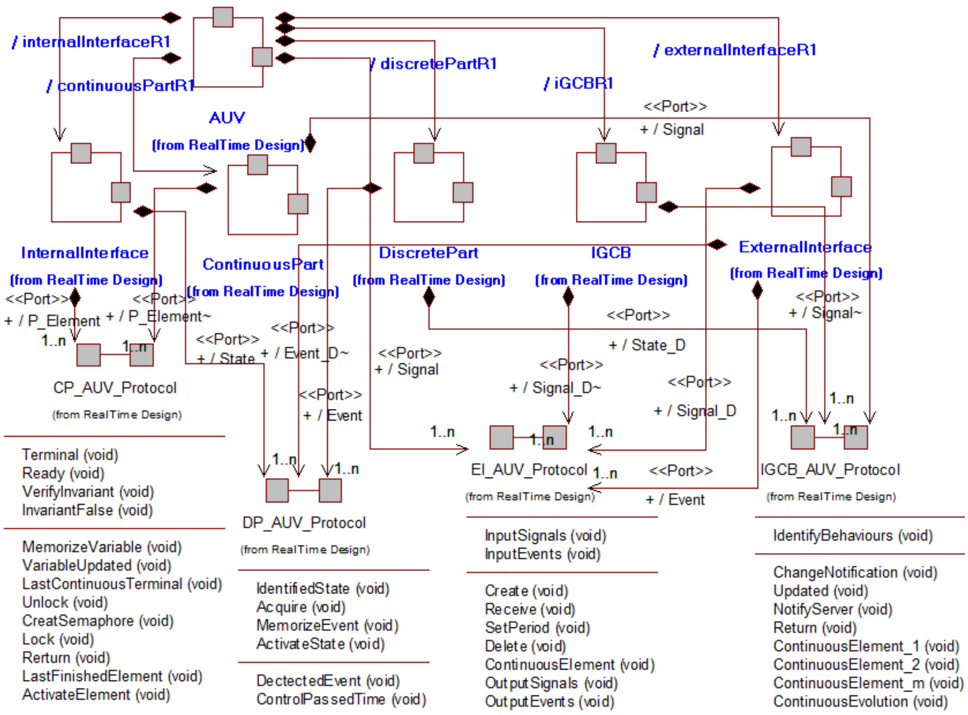

Figure 6. Structure of control capsules for the developed AUV.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 July 2024 doi:10.20944/preprints202407.0154.v1

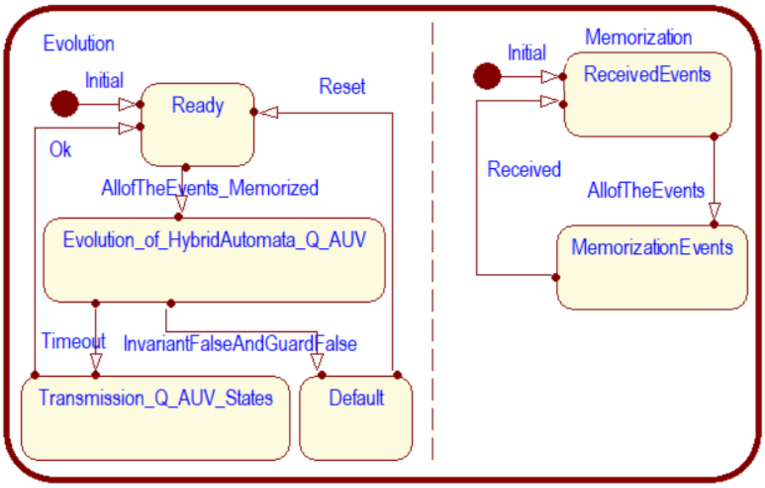

Figure 7. The local state machine in the capsule of *discrete part* in HA evolution.

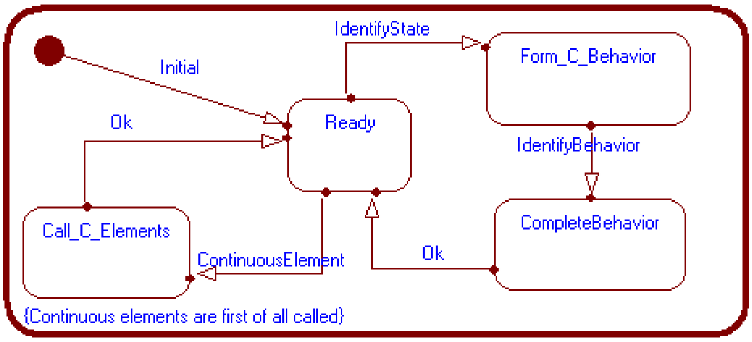

Figure 8. The local state machine in the capsule of *IGCB* for the HA evolution.

As mentioned in Section 1, the ability to reuse plays an important role in developing a control system as it mitigates time, cost, and resources. In the cycle of controller development, the reusability can be expanded by the control capsule-based pattern and its components, e.g., the protocols, ports, connectors, internal structures, and local state machines described in Figures 5-8.

3.3. PSM Implementation for an AUV Controller

For realizing AUV controllers, PIM is transformed into a model of implementation. This process can be performed by using various object-oriented software platforms, such as C++, C#, Java, which support completely performing the model on industrial-specific platforms (EPC, MCU, PLC). The ultimate-designed model by PIM is first tested by the simulation model. There are various software tools that can perform this simulation, such as IBM Rational Rose Real-time [67] and Open-Modelica [68]. From the simulation results, we can estimate the performance of the designed controller. In the next step, the design artifacts of the model can be optimized before realization. These optimized elements, after simulation, are utilized to adapt PIM to get the updated design model. In the final step, the optimized PIM is converted into a deployment model embedded on available and compatible hardware via IDE tools. Here, round-trip engineering is used to perform the transformations. These techniques are, for example, backward and forward techniques in intermediate programming, with responses of 80% and 20%, respectively, for generated codes and handcrafted codes.

4. Application

On the basis of this new model, a controller for tracking a surface trajectory is designed and then deployed on the hardware platform. This controller manipulates a small-scale T-AUV (turtle-shaped

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 2 July 2024 doi:10.20944/preprints202407.0154.v1

autonomous underwater vehicle) to track a desired trajectory on the water surface. The functions of the T-AUV are shown in Table 3.

According to the AUV dynamic models described in Equations (1)–(2), in combination with the assumption that the vertical and lateral motions are negligible because the T-AUV possesses a turtle shape, the dynamic model of the T-AUV can only be concerned with the longitudinal axis line.

Table 3. Key specifications of the T-AUV.

Specifications	Value
Size(L×W×H)	(1.26×0.61×0.40)m
operating time	20min
Dry weight	21.20kg
2×Li-Po battery	22.2V, 20000mAh
Max. capacity	324W
Max. submerged depth	1.20m
Max. radius of operation	450m
Max. submerging or rising speed	0.30m/s
Max. horizontal speed	1.80m/s

The UKF algorithm (Algorithm 1) is mapped to sensors attached to the T-AUV. These sensors can be the GPS Ublox Neo 6M [69] and the IMU MPU6000 [70]. In this study, MCUs of types ATMEGA32-U2 and STM32 Cortex-M4 [71] were utilized to embed the control program for the AUV controller. The whole physical M-AUV was built in order to test the controller, as shown in Figure 9. For a practical test, we predefined different course angles, trajectories, and velocities. The trial results are displayed in Table 4. Figure 10a–b illustrate the trial results, where the T-AUV is controlled to follow given triangle-shaped and rectangle-shaped trajectories with an average velocity of 1.5 m/s, respectively. Tables 5 and 6 show the trajectory-tracking errors based on the RMS (root mean square) calculation as well as peak values.

Figure 9. M-AUV real model and trial cruises

Table 4. The test data of the horizontal planar course tracking.

№	Predetermined course angle (deg)	Average velocity (m/s)	Time of stabilized course (s)
1	10	1.0	6.90
2	10	1.5	6.10
3	20	1.0	7.20
4	20	1.5	6.40
5	30	1.0	7.30
6	30	1.5	7.10

Table 5. The path tracking error of the T-AUV model followed up the predetermined planar triangle-shaped trajectory.

WPs-based path generation	RMS deviation (m)		Max deviation (m)	
	Along east axis	Along north axis	Along east axis	Along north axis
WP0 - WP1	3.51552	3.82270	4.07784	4.24404
WP1 - WP2	2.60232	3.34176	4.06368	4.30740
WP2 - WP0	2.35188	5.01288	2.79348	7.67352

Table 6. The path tracking error of the T-AUV followed up the predetermined planar rectangle-shaped trajectory.

WPs-based path generation	RMS deviation (m)		Max deviation (m)	
	Along east axis	Along north axis	Along east axis	Along north axis
WP0 - WP1	2.7036	3.0876	2.9207	3.4398
WP1 - WP2	1.7465	2.1935	2.3279	2.8723
WP2 - WP3	2.0754	2.6527	2.2675	2.0894
WP3 - WP4	2.0532	2.0921	2.2793	2.7260
WP4 - WP0	1.7270	1.5573	1.9874	1.7674

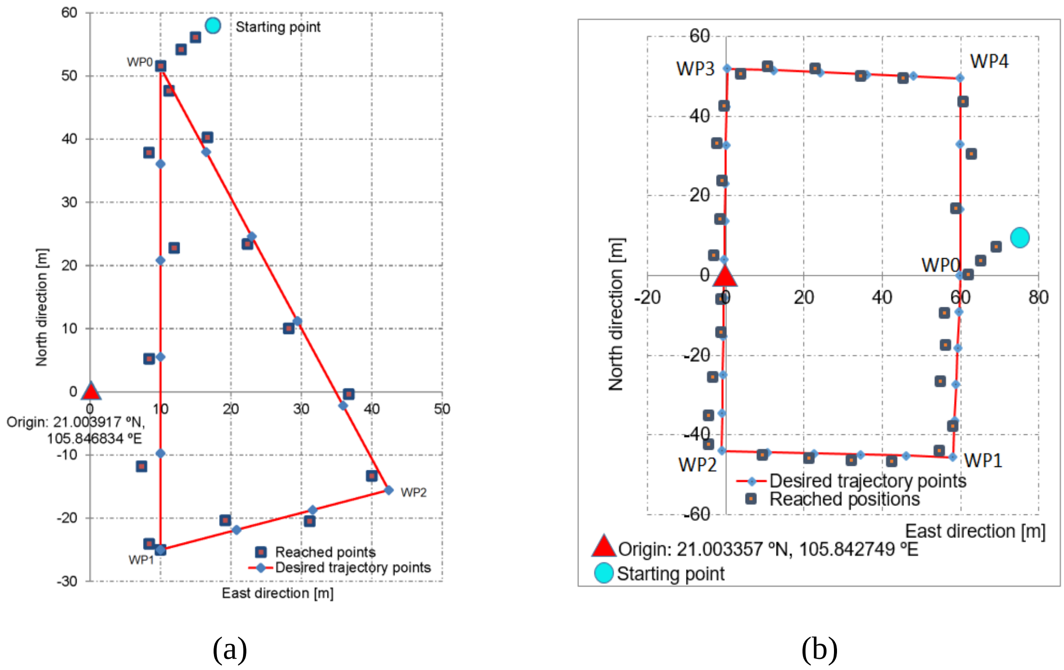

Figure 10. T-AUV approached and followed the triangle-shaped trajectory (**a**) and rectangle-shaped trajectory (**b**).

In comparison to the previous experimental results of the author's research [61] with PID control and the same AUV model, the new proposed controller improves the tracking error and time stabilization, which show a drop of 1.5% and 5.5%, respectively. For this new controller, the IB control law and UKF implementations were combined to build the control blocks for depth, attitude, and position. It is shown that the new proposed controller, accompanied by IB and UKF techniques, improves the control performance over using a single PID regulator.

5. Conclusions and Future Work

This paper introduces a hybrid control model that effectively supports deploying control parts of AUVs. The proposed model is followed by OOSEM specialization combined with MDA concepts, the RealTime UML/SysML, Hybrid Automata, and the UKF algorithm to closely deploy AUV controllers. The dynamics and control architecture of AUV were first used for the inputs of the development lifecycle and integrated with the OOSEM/MDA's features (CIM, PIM, and PSM). In CIM, we define the use case model, which is then combined with continuous models such as the FBD (supplemented functional block) diagram and HA for capturing control requirements in detail. The PIM was equipped with the UKF algorithm to create the design pattern of the control capsules. Next, the designed PIM was transformed into a PSM that deployed the controller with compatible hardware. Finally, a controller for planar trajectory-tracking dedicated to a real model of T-AUV was implemented and tested with trial cruises. Following the above OOSEM/MDA specialization and the RealTime UML/SysML, system developers can handle complex systems by utilizing developed model artifacts and traceability in a visual way. Nevertheless, as the RealTime UML/SysML version does not have notations to model complex continuous evolutions combined with control laws in developing control systems, a supplemented FBD could thus be added to the CIM of the developed AUV.

In the future, we plan to develop this proposed framework and application equipped with compatible physical sensors and industrial microcontrollers to create a control system that could permit the M-AUV to be used for monitoring and patrolling the marine environment of our country (Vietnam).

Author Contributions: Conceptualization, C.D.S., N.V.H., N.V.H.; methodology, C.D.S, N.V.H., N.V.H. and N.T.K.; software, C.D.S, N.V.H., N.V.H..; validation, C.D.S, N.V.H., N.V.H.; writing–original draft preparation, C.D.S, N.V.H., N.V.H. and N.T.K.; writing—review and editing C.D.S, N.V.H., N.V.H. and N.T.K.; supervision, N.V.H.; project administration, N.V.H. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: The authors would like to thank the supports of the Vietnam Ministry of Education and Training under grant number B2023-BKA-13.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. S. Sivčev, J. Coleman, E. Omerdić, G. Dooly, D. Toal, Underwater manipulators: A review, Ocean Engineering, Elsevier, ISSN 0029-8018, 163 (2018), 431-450. https://doi.org/10.1016/j.oceaneng.2018.06.018
2. Y.R. Petillot, G. Antonelli, G. Casalino, F. Ferreira, Underwater Robots: From Remotely Operated Vehicles to Intervention-Autonomous Underwater Vehicles, IEEE Robotics & Automation Magazin, ISSN 1070-9932, 26 (2019), 94-101. https://doi.org/10.1109/MRA.2019.2908063
3. J. Bao, D. Li, X. Qiao, T. Rauschenbach, Integrated navigation for autonomous underwater vehicles in aquaculture: A review, Information Processing in Agriculture, Elsevier, ISSN 2214-3173, 7 (2020), 139-151. https://doi.org/10.1016/j.inpa.2019.04.003
4. AUVAC. Autonomous Undersea Vehicles Applications Center. Available: https://auvac.org/, 2022 (accessed February 2022).
5. T.A. Henzinger, P.W. Kopke, A. Puri, P. Varaiya, What's Decidable about Hybrid Automata?, Journal of Computer and System Sciences, Elsevier, ISSN 0022-0000, 57 (1998), 94-124. https://doi.org/10.1006/jcss.1998.1581
6. L.P. Carloni, R. Passerone, A. Pinto, V.A. Sangiovanni, Languages and Tools for Hybrid Systems Design, Now Publishers Inc, Boston, 2006.
7. P.A. Fishwick, Handbook of Dynamic System Modeling, Taylor & Francis Group, USA, 2007.

8. K. El Hamidi, M. Mjahed, A. El Kari, H. Ayad, Neural Network and Fuzzy-logic-based Self-tuning PID Control for Quadcopter Path Tracking, Studies in Informatics and Control, ISSN 1220-1766, 28 (2019), 401-412. https://doi.org/10.24846/v28i4y201904
9. S. Gao, R. Song, Y. Zheng, Y. Li, Robust Coordinated Tracking Control of Multiple Robots System Under Bounded Inputs, Studies in Informatics and Control, ISSN 1220-1766, 29 (2020), 283-292. https://doi.org/10.24846/v29i3y202002
10. A.Y. Ouadine, M. Mjahed, H. Ayad, A. El Kari, UAV Quadrotor Fault Detection and Isolation Using Artificial Neural Network and Hammerstein-Wiener Model, Studies in Informatics and Control, ISSN 1220-1766, 29 (2020), 317-328. https://doi.org/10.24846/v29i3y202005
11. E. Dolicanin, I. Fetahovic, E. Tuba, R. Capor-Hrosik, M. Tuba, Unmanned Combat Aerial Vehicle Path Planning by Brain Storm Optimization Algorithm, Studies in Informatics and Control, ISSN 1220-1766, 27 (2018), 15-24. https://doi.org/10.24846/v27i1y201802
12. H.R. Karimi, A sliding mode approach to H∞ synchronization of master–slave time-delay systems with Markovian jumping parameters and nonlinear uncertainties, Journal of the Franklin Institute, Elsevier, ISSN 0016-0032, 349 (2012), 1480-1496. https://doi.org/10.1016/j.jfranklin.2011.09.015
13. H. Nawaz, H. Mansoor Ali, Implementation of Cross Layer Design for Efficient Power and Routing in UAV Communication Networks, Studies in Informatics and Control, ISSN 1220-1766, 29 (2020), 111-120. https://doi.org/10.24846/v29i1y202011
14. K. Shi, Z. Wu, B. Jiang, H.R. Karimi, Dynamic path planning of mobile robot based on improved simulated annealing algorithm, Journal of the Franklin Institute, Elsevier, ISSN 0016-0032, In Press (2023), 32 pages. https://doi.org/10.1016/j.jfranklin.2023.01.033
15. H.R. Karimi, How to deal with the complexity in robotic systems?, Complex Engineering Systems, ISSN 2770-6249, 2 (2022), 3 pages. https://doi.org/10.20517/ces.2022.33
16. S.K. Valluru, M. Kaur, K. Kartikeya, A. Goel, D. Dobhal, Experimental Investigation of Fully Informed Particle Swarm Optimization tuned Multi Loop L-PID and NL-PID Controllers for Gantry Crane System, Procedia Computer Science, Elsevier, ISSN 1877-0509, 171 (2020), 130-138. https://doi.org/10.1016/j.procs.2020.04.014
17. J. Guerrero, J. Torres, V. Creuze, A. Chemori, E. Campos, Saturation based nonlinear PID control for underwater vehicles: Design, stability analysis and experiments, Mechatronics: The Science of Intelligent Machines, Elsevier, ISSN 0957-4158, 61 (2019), 96-105. https://doi.org/10.1016/j.mechatronics.2019.06.006
18. L. Liu, L. Zhang, G. Pan, S. Zhang, Robust yaw control of autonomous underwater vehicle based on fractional-order PID controller, ID 111493, Ocean Engineering, Elsevier, ISSN 0029-8018, 257 (2022), 8 pages. https://doi.org/10.1016/j.oceaneng.2022.111493
19. A. Alaeddini, K.A. Morgansen, M. Mesbahi, Augmented state feedback for improving observability of linear systems with nonlinear measurements, Systems & Control Letters, Elsevier, ISSN 0167-6911, 133 (2019), 8 pages. https://doi.org/10.1016/j.sysconle.2019.104520
20. A.A.R.A. Makdah, N. Daher, D. Asmar, E. Shammas, Three-dimensional trajectory tracking of a hybrid autonomous underwater vehicle in the presence of underwater current, Ocean Engineering, Elsevier, ISSN 0029-8018, 185 (2019), 115-132. https://doi.org/10.1016/j.oceaneng.2019.05.030
21. M. Lei, Nonlinear diving stability and control for an AUV via singular perturbation, Ocean Engineering, Elsevier, ISSN 0029-8018, 197 (2020), 11 pages. https://doi.org/10.1016/j.oceaneng.2019.106824

22. A.K. Khalaji, H. Tourajizadeh, Nonlinear Lyapounov based control of an underwater vehicle in presence of uncertainties and obstacles, Ocean Engineering, Elsevier, ISSN 0029-8018, 198 (2020), 9 pages. https://doi.org/10.1016/j.oceaneng.2020.106998
23. J. Liu, M. Zhao, L. Qiao, Adaptive barrier Lyapunov function-based obstacle avoidance control for an autonomous underwater vehicle with multiple static and moving obstacles, ID 110303, Ocean Engineering, Elsevier, ISSN 0029-8018 243 (2022), 16 pages. https://doi.org/10.1016/j.oceaneng.2021.110303
24. G.R. Cho, D.G. Park, H. Kang, M.J. Lee, J.H. Li, Horizontal Trajectory Tracking of Underactuated AUV using Backstepping Approach, IFAC-PapersOnLine, Elsevier, ISSN 2405-8963, 52 (2019), 174-179. https://doi.org/10.1016/j.ifacol.2019.11.774
25. P. Du, W. Yang, Y. Wang, R. Hu, Y. Chen, S.H. Huang, A novel adaptive backstepping sliding mode control for a lightweight autonomous underwater vehicle with input saturation, ID 112362, Ocean Engineering, Elsevier, ISSN 0029-8018, 263 (2022), 12 pages. https://doi.org/10.1016/j.oceaneng.2022.112362
26. Z. Yan, M. Wang, J. Xu, Robust adaptive sliding mode control of underactuated autonomous underwater vehicles with uncertain dynamics, Ocean Engineering, Elsevier, ISSN 0029-8018, 173 (2019), 802-809. https://doi.org/10.1016/j.oceaneng.2019.01.008
27. B. Su, H.B. Wang, Y. Wang, Dynamic event-triggered formation control for AUVs with fixed-time integral sliding mode disturbance observer, ID 109893, Ocean Engineering, Elsevier, ISSN 0029-8018 240 (2021), 14 pages. https://doi.org/10.1016/j.oceaneng.2021.109893
28. J. Guerrero, J. Torres, V. Creuze, A. Chemori, Adaptive disturbance observer for trajectory tracking control of underwater vehicles, Ocean Engineering, Elsevier, ISSN 0029-8018, 200 (2020), 13 pages. https://doi.org/10.1016/j.oceaneng.2020.107080
29. J. Zhou, X. Zhao, T. Chen, Z. Yan, Z. Yang, Trajectory Tracking Control of an Underactuated AUV Based on Backstepping Sliding Mode With State Prediction, IEEE Access, ISSN 2169-3536, 7 (2019), 181983-181993. https://doi.org/10.1109/ACCESS.2019.2958360
30. O. Elhaki, K. Shojaei, A robust neural network approximation-based prescribed performance output-feedback controller for autonomous underwater vehicles with actuators saturation, Engineering Applications of Artificial Intelligence, Elsevier, ISSN 0952-1976, 88 (2020), 16 pages. https://doi.org/10.1016/j.engappai.2019.103382
31. N. Kumar, M. Rani, An efficient hybrid approach for trajectory tracking control of autonomous underwater vehicles, Applied Ocean Research, Elsevier, ISSN 0141-1187, 95 (2020), 10 pages. https://doi.org/10.1016/j.apor.2020.102053
32. K. Fang, H. Fang, J. Zhang, J. Yao, J. Li, Neural adaptive output feedback tracking control of underactuated AUVs, ID 109211, Ocean Engineering, Elsevier, ISSN 0029-8018 234 (2021), 11 pages. https://doi.org/10.1016/j.oceaneng.2021.109211
33. D. Wang, Y. Shen, J. Wan, Q. Sha, G. Li, G. Chen, B. He, Sliding mode heading control for AUV based on continuous hybrid model-free and model-based reinforcement learning, Applied Ocean Research, Elsevier, ISSN 0141-1187, 118 (2022), 14 pages. https://doi.org/10.1016/j.apor.2021.102960
34. N. Wang, T. Chen, S. Liu, R. Wang, H.R. Karimi, Y. Lin, Deep Learning-based Visual Detection of Marine Organisms: A Survey, Neurocomputing, Elsevier, ISSN 1872-8286, In Press (2023), 38 pages. https://doi.org/10.1016/j.neucom.2023.02.018
35. B. Hadi, A. Khosravi, P. Sarhadi, Deep reinforcement learning for adaptive path planning and control of an autonomous underwater vehicle, Aplied Ocean Research, Elsevier, ISSN 0141-1187, 129 (2022), 14 pages. https://doi.org/10.1016/j.apor.2022.103326

36. OMG, Documents Associated With Unified Modeling Language™ (UML® Version 2.5.1): OMG formal/17-12-05, OMG, http://www.omg.org/spec/UML/, 2017.
37. OMG, SysML Specifications Version 1.6: OMG formal/19-11-01, OMG, https://www.omg.org/spec/SysML/, 2019.
38. H. Lykins, S. Friedenthal, A. Meilich, Adapting UML for an Object Oriented Systems Engineering Method (OOSEM), Proc INCOSE Int Symp, July 16–20, INCOSE, Minneapolis, MN, 2000, pp. 490-497. https://doi.org/10.1002/j.2334-5837.2000.tb00416.x
39. P. Pearce, M.C. Hause, ISO-15288, OOSEM and Model-Based Submarine Design, the 6th Asia Pacific Conference on Systems Engineering, Deep Blue Tech, Brisbane, Australia, 2012, pp. 15 pages.
40. INCOSE. Object-Oriented SE Method. Available: https://www.incose.org/incose-member-resources/working-groups/transformational/object-oriented-se-method, 2020 (accessed September 2020).
41. INCOSE, Systems Engineering Vision 2025, INCOSE, San Diego, CA 92111-2222, USA, 2014.
42. INCOSE. Model-Based Systems Engineering (MBSE). Available: https://www.incose.org/, 2022 (accessed January 2022).
43. OMG, Model Driven Architecture (MDA): Guide revision 2.0 of MDA Guide Version 1.0.1 (12th June 2003). OMG Document ormsc/2014-06-01, OMG, http://www.omg.org/cgi-bin/doc?ormsc/14-06-01, 2014.
44. OMG. MDA Success Stories. Available: https://www.omg.org/mda/products_success.htm, 2022 (accessed June 2022).
45. L.T.W. Agner, I.W. Soares, P.C. Stadzisz, J.M. Simão, A Brazilian survey on UML and model-driven practices for embedded software development, Systems and Software, Elsevier, ISSN 0164-1212, 86 (2013), 997–1005. https://doi.org/10.1016/j.jss.2012.11.023
46. M. Rashid, M.W. Anwar, A.M. Khan, Toward the tools selection in model based system engineering for embedded systems—A systematic literature review, Journal of Systems and Software, Elsevier, ISSN 0164-1212, 106 (2015), 150-163. https://doi.org/10.1016/j.jss.2015.04.089
47. L.O. Freire, L.M. Oliveira, R.T.S. Vale, M. Medeiros, R.E.Y. Diana, R.M. Lopes, E.L. Pellini, E.A. Barros, Development of an AUV control architecture based on systems engineering concepts, Ocean Engineering, Elsevier, ISSN 0029-8018, 151 (2018), 157-169. https://doi.org/10.1016/j.oceaneng.2018.01.016
48. N.V. Hien, N.V. He, P.G. Diem, A model-driven implementation to realize controllers for Autonomous Underwater Vehicles, Applied Ocean Research, Elsevier, ISSN 0141-1187, 78 (2018), 307-319. https://doi.org/10.1016/j.apor.2018.06.020
49. T. Soriano, N.V. Hien, K.M. Tuan, T.V. Anh, An object-unified approach to develop controllers for autonomous underwater vehicles, Mechatronics: The Science of Intelligent Machines, Elsevier, ISSN 0957-4158, 35 (2016), 54-70. https://doi.org/10.1016/j.mechatronics.2015.12.011
50. M.W. Anwar, M. Rashid, F. Azam, M. Kashif, Model-based design verification for embedded systems through SVOCL: an OCL extension for SystemVerilog, Design Automation for Embedded Systems, Springer, ISSN 0929-5585, 21 (2017), 1-36. https://doi.org/10.1007/s10617-017-9182-z
51. M.W. Anwar, M. Rashid, F. Azam, M. Kashif, W.H. Butt, A model-driven framework for design and verification of embedded systems through SystemVerilog, Design Automation for Embedded Systems, Springer, ISSN 0929-5585, 23 (2019), 179–223. https://doi.org/10.1007/s10617-019-09229-y
52. T. Soriano, H.A. Pham, N.V. Hien, Analysis of coordination modes for multi-UUV based on Model Driven Architecture, 12th France-Japan and 10th Europe-Asia Congress on Mechatronics, IEEE, Tsu, Japan, 2018. https://doi.org/10.1109/MECATRONICS.2018.8495893

53. OMG, UML Profile for MARTE: UML for model-driven development of Real Time and Embedded Systems (RTES), OMG formal/19-04-02, https://www.omg.org/spec/MARTE/, OMG, 2019.
54. B.P. Douglass, Real-Time UML Workshop for Embedded Systems (2nd Edition), Elsevier, Oxford, UK, 2014.
55. B. Selic, S. Gerard, Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE, Elsevier, USA, 2014.
56. B. Selic, Using UML for modeling complex real-time systems, Lecture Notes in Computer Science, Springer, ISSN 0302-9743, 1474 (1998), 250-260. https://doi.org/10.1007/BFb0057795
57. SNAME, Nomenclature for Treating the Motion of a Submerged Body through a Fluid, SNAME, New York 18, N. Y., USA, 1950.
58. T.I. Fossen, Handbook of Marine Craft Hydrodynamics and Motion Control, John Wiley & Sons, United Kingdom, 2011.
59. Y. Bar-Shalom, X.R. Li, T. Kirubarajan, Estimation with Applications to Tracking and Navigation- Theory Algorithms and Software, John Wiley & Sons, USA, 2001.
60. B. Allotta, A. Caitib, R. Costanzi, F. Fanelli, D. Fenucci, E. Meli, A. Ridolfi, A new AUV navigation system exploiting unscented Kalman filter, Ocean Engineering, Elsevier, ISSN 0029-8018, 113 (2016), 121–132. https://doi.org/10.1016/j.oceaneng.2015.12.058
61. N.V. Hien, N.V. He, V.T. Truong, P.G. Diem, Using the Real-Time Unified Modeling Language to Implement an AUV Controller. Research project report, funded by State of Vietnam, KC03.TN05/11-15, Hanoi University of Science and Technology, Hanoi, Vietnam, 2013.
62. P. Sarhadi, A.R. Noei, A. Khosravi, Model reference adaptive PID control with anti-windup compensator for an autonomous underwater vehicle, Robotics and Autonomous Systems, Elsevier, ISSN 0921-8890, 83 (2016), 87-93. https://doi.org/10.1016/j.robot.2016.05.016
63. F. Kong, Y. Guo, W. Lyu, Dynamics Modeling and Motion Control of an New Unmanned Underwater Vehicle, IEEE Access, ISSN 2169-3536, 8 (2020), 30119-30126. https://doi.org/10.1109/ACCESS.2020.2972336
64. [64] J. Wan, B. He, D. Wang, T. Yan, Y. Shen, Fractional-Order PID Motion Control for AUV Using Cloud-Model-Based Quantum Genetic Algorithm, IEEE Access, ISSN 2169-3536, 7 (2019), 124828-124843. https://doi.org/10.1109/ACCESS.2019.2937978
65. X. Wang, G. Zhang, Y. Sun, J. Cao, L. Wan, M. Sheng, Y. Liu, AUV near-wall-following control based on adaptive disturbance observer, Ocean Engineering, Elsevier, ISSN 0029-8018, 190 (2019), 17 pages. https://doi.org/10.1016/j.oceaneng.2019.106429
66. Z. Yan, Z. Yang, J. Zhang, J. Zhou, A. Jiang, X. Du, Trajectory Tracking Control of UUV Based on Backstepping Sliding Mode With Fuzzy Switching Gain in Diving Plane, IEEE Access, ISSN 2169-3536, 7 (2019), 166788-166795. https://doi.org/10.1109/ACCESS.2019.2953530
67. IBM. IBM Rational's Methodology, Software, Online Documentation and Training Kits. Available: https://www.ibm.com/academic/home, 2022 (accessed April 2022).
68. OpenModelica. OpenModelica. OpenModelica software, version 1.18. Available: https://www.openmodelica.org/, 2022 (accessed February 2022).
69. u-blox. Product selector. Available: https://www.u-blox.com/en/product-search, 2022 (accessed January 2022).

 doi:10.20944/preprints202407.0154.v1

70. InvenSense. Sensor System on Chip. Available: http://www.invensense.com/, 2022 (accessed January 2022).
71. Arduino. Open-source electronics prototyping platform for hardware and software. Available: http://www.arduino.cc/, 2022 (accessed March 2022).