# Software architecture for interactive robot teleoperation

Nader Cheaib, Mouna Essabbah, Christophe Domingues, Samir Otmane

# Software architecture for interactive robot teleoperation

**Nader Cheaib, Mouna Essabbah, Christophe Domingues, Samir Otmane**
IBISC Laboratory, University of Evry Val d'Essonne
40 Pelvoux Street, 91020 Evry, France
firstname.lastname@ibisc.fr

**ABSTRACT**

In this paper, we present a software architecture for interactive and collaborative underwater robot teleoperation. This work is in the context of the Digital Ocean Europe project that aims at digitalizing seafloor sites in 3D imagery using underwater robots (ROVs), and uses this information in order to edit interactive, virtually animated environments diffused online. The work presented in this paper concerns the software architecture of the interactive system in order to collaboratively teleoperate the robot, using two types of interfaces: 1) an intuitive web interface and 2) a Virtual Reality (VR) platform. The particularity of our system is the separation of the systems' functional core from its interfaces, which enables greater flexibility in teleoperating the robot. We discuss the conceptual software architecture as well as the implementation of the systems' interfaces.

**Keywords**

Interactive systems, Software Architecture, CSCW, Virtual Reality.

**INTRODUCTION**

Teleoperation consists of remotely commanding and manipulating robot systems. This type of control allows doing complex tasks in hostile environments, where some of them may be hard to accomplish by humans. The application domains of teleoperation are numerous and present in most of research fields (medical, spatial, etc.). In particular, remote operation of underwater robotic systems seems to be a growing research concern in many application domains [12, 14]. On the other hand, as the use of the internet and the services offered with it are exponantially emerging, people are in increasing need of flexible and agile applications in order to execute collaborative tasks. The emergence of collaborative work over the internet was a solution to the high complexity of systems and the technical difficulties that arise from their use, as geographically distributed users are increasingly working together on common tasks, but using rigid and often incompatible applications. In this paper, we propose a new groupware (collaborative software) architecture to collaboratively teleoperate an underwater robot, independently of the interfaces used. Hence, people connected to the internet are able to teleoperate the robot using a PC, mobile or virtual reality platforms. This work is in the context of the Digital Ocean Europe that aims at enhancing public awareness on the ocean and increases their marine scientific literacy. Hence, the aim is to give public means to remotely operate an underwater robot online in order to to discover underwater environments.

We will proceed as follows: Firstly, we present some related work in the field of underwater robot teleoperation. Then, we define the concept of collaboration and the need for a new software architecture supporting it. Then, we present our conceptual software architecture. After that, we describe two types of interfaces that our software architecture supports for robot teleoperation: an intuitive web interface and a VR platform. We discuss the originality of our system as well as its application to robot teleoperation. Finally, we present a conclusion and perspectives in the field.

**RELATED WORK**

Many researchers have proposed software architectures for teleoperating underwater vehicles. The authors in [17] describe a system for long-distance remote observation of robotic operations targeted to e-learning, called AQUA. They present a software architecture that provides a uniform look-and-feel web interface presenting sensor information on the distance robot. Other work, such as in [18], present a system that facilitates interactive remote control of a high definition camera on an underwater robot, while transmitting the video feedback using web services. The aim of this system is to enable the public to control their own view of the undersea environment, independently of their location. The authors in [4] present the E-Robot project that enables users to interact with an underwater ROV (Remotely Operated Vehicle) through an Internet Browser to pilot the ROV in real time, while visualizing underwater images taken under the ice in the Arctic region.

Furthermore, the authors in [13] present an Internet-operated deep-sea crawler, equipped with sensors to measure the temperature, pressure, water currents, methane and turbidity. This system, called Wally, supports a pan/tilt webcam, affording detailed views of the seafloor sediments and local sea life. The authors in [2] present GOYA, a

teleoperated system for blasting applied to hull cleaning in ship maintenance. The authors followed the COMET methodology for designing the systems' classes in order to design the robots' control units. Finally, the authors in [1] present a reference architecture for robot teleoperation systems developed using the domain-engineering process and architectural patterns. This software architecture has been applied to various systems for maintenance activities in nuclear power plants, such as the ROSA, IRV and TRON systems.

In fact, a common aspect for most of these systems is their use of web interfaces for robot teleoperation. However, they do not take in consideration other types of interfaces, as well as collaborative aspects of teleoperation. For the authors in [11], user-interfaces' design for teleoperation involves a trade-off between ease of use and the capacity for complex tasks. This is a challenge for web-based interfaces as they need to support users having diverse skills. Hence, web interfaces should be designed so that novice users feel comfortable using it, while not being a constraint on expert users. Also, the authors in [15] present user interface issues to consider while designing collaborative teleoperation, such as visible navigation aids, chat channels, data presentation, etc. In this paper, we present a software architecture that tries to remedy these constraints by separating the functional core of the system from its physical interfaces. It also takes in consideration collaboration between users, by dividing the collaborative experience into communication, coordination and production spaces.

## COLLABORATION AND THE 3C MODEL

In order to further understand the concept of collaboration, we base our work on the 3C functional model proposed by Ellis [8], shown in Figure 1. According to this model, a groupware system covers three domain specific functions, production/cooperation, communication and coordination. The production space designates the objects resulting from the activity of the group (ex: word document, paint etc.). For Ellis, this space is concerned with the result of common tasks to be achieved, and is the space where the production takes place. The coordination space defines the actors and their social structure, as well as different tasks to be accomplished in order to produce in the production space.

Ellis eventually completed the model with the communication space that offers to actors in the coordination space means to exchange information in which the semantics concern exclusively the actor, and where the system only acts as a messenger. In our work, we use this decomposition of groupware's functionalities in order to introduce a collaborative software architecture supporting the functional decomposition of services that can be present in an interactive groupware system.

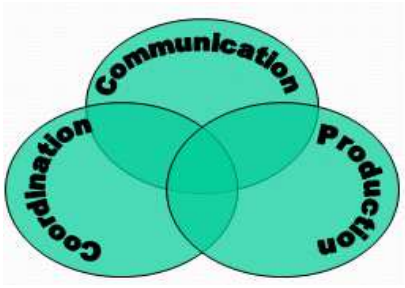


Figure 1: 3C model by Ellis

In fact, there exist some work in the literature that make use of the 3C model in order to create collaborative applications [10, 16, 20] for various application domains. The authors in [10] affirm that the understanding of communication has transformed from being vertical, where orders are passed from above and reports sent up the line, to a peer-to-peer paradigm where communication, coordination and cooperation predominate. This is due to the fact that command and control paradigm is losing effectiveness in the society. People are increasingly using tools and applications with no specific or centralized source that issues orders, but where people are collaboratively coordinating and dividing tasks between them, and eventually taking group decisions. In our work, we use the 3C model to define the three main aspects of a collaborative application. Hence, an optimal collaboration pattern is achieved when the collaborative process is initiated by communicating, and ends by a concrete realization of the task at hand.

## GENERIC SOFTWARE ARCHITECTURE

We rely on the Arch model [3] that separates the physical interface (Layer 0 in Figure 2) from the Functional Core (FC) of the system (Layers N-1 and N). However, in contrast to the Arch model where the FC is a dead-end component (implements static domain functionalities), our FC is connected to the internet in order to communicate with the external world (Internet). In this paper, we discuss the FC's design (Layers N-1 and N) as well as the physical layer (Layer 0). Furthermore, we rely on Dewan's model [7], which is a generalization of the Arch model. This model structures a groupware system into a variable number of replicated and shared layers. Thus, it defines a collaboration degree between the system's components and users, where the highest layer is the most semantic one, corresponding to the FC of the system (coincides with the one of the Arch model as well as the Abstraction facet of the PAC* model [5]), and the lowest layer representing the material level (corresponds to the Arch's Physical Interaction component as well as the Presentation facet of PAC*). Note that Figure 2 representing our proposed architecture shows only the FC of the system, along with the physical interaction layer that implements interactions between users.

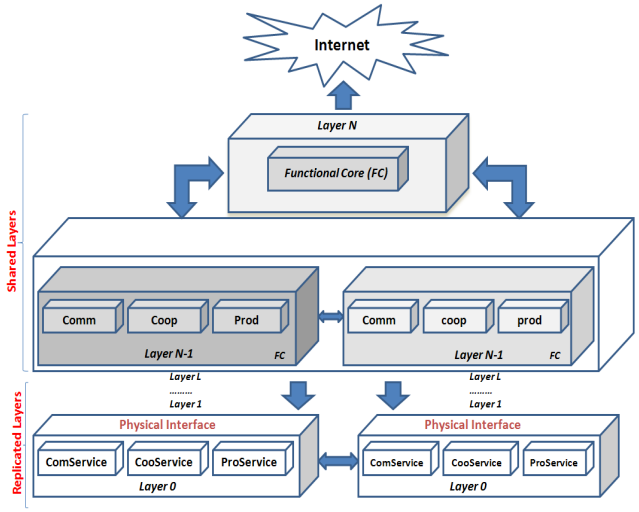


Figure 2: Conceptual software architecture

Our software architecture is constituted of a root representing shared layers among all users in the system, as well as several replicated layers for every user. The layers communicate vertically using interaction events, and horizontally through collaboration between users over the network. However, in contrast to the Clover model [20] where the functional core is also split into two layers: one private and shared, while the other is replicated and public, our functional core is represented by two layers that are both shared and constitute the root of the system.

### Functional Core (FC)

The shared layers of the software architecture constituting the FC enable users to manipulate domain objects and have access to various services in the system, while the replicated layers handles the set of services and the state of the system that is private for every user in collaboration. We extend this layer abstraction as in [20] by decomposing each layer of the architecture into sub-components, each dedicated to one facet of Ellis' 3C model, while providing and managing specific services for communication, coordination and production on the layer N-1. These services can be considered as orchestrations of atomic services based on the functionalities they offer, and are exposed to users through systems' interfaces. In our work, we suppose that only the layers on the level N-1 and on the lowest level (Layer 0) satisfy these three main classifications, while we make no assumption about the decomposition of the highest layer in the software architecture, which is mainly composed of a module to synchronize data from users collaborating using the system. Further information concerning the software architecture can be found in our earlier work [6].

## INTERACTIVE INTERFACES FOR ROV TELEOPERATION

We present two types of Human-Robot Interfaces (HRI) hosted on the layer 0 of our software architecture. Our first HRI enables an easy access to teleoperate the ROV via a simple web browser. In our project, we have integrated a Web interface on a submersible device called Dolphyn, shown in Figure 3, which is an aquatic PC integrating an x86 tablet running on Windows 7. This device diffuses multimedia content while using two integrated joysticks to teleoperate the ROV via Internet. It aims at visualizing underwater media while being in a swimming pool for a more realistic underwater exploration. The second HRI is based on a VR/Augmented Reality (AR) platform, which gives users a multisensory exploration of the underwater site. Hence, it enhances the feeling of presence due to stereoscopic display and haptic interfaces.

In fact, three main components are used in our system: 1) The client side application (Web or VR/AR) on the Layer 0, 2) the ROV's server and the video streaming service to control the distant ROV while capturing video images on the Layer N-1, and 3) the multiuser service on the Layer N. Recall that Layer N-1 and Layer N represent the system's FC. In our case, the user interacts with the system through its interfaces in order to visualize the content diffused by ROV's camera, while using the Dolphyn.

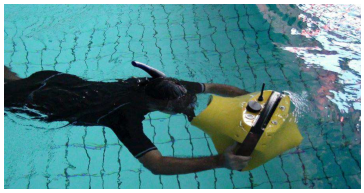

Figure 3: The Dolphyn

Indeed, the real-time streaming service allows bringing the video broadcast to the internet. The process involves a camera on the ROV, an encoder to digitalize the content as well as a content delivery network in order to distribute and deliver the content. The media can then be viewed by end-users in real time. For encoding the PAL signal, we have chosen the Ogg format, where we use the HTTP protocol for delivery. On the other hand, the multiuser service is used to synchronize data between users performing a collaborative virtual diving. Indeed, Layer N of our software architecture is used to accept network connections and transfer commands sent by users to the ROV. It is also used to prevent multiple and heavy connections to the ROV through the use of a priority list (First come first served).

### Web HRI on the 3C Model

We present the Web HRI dedicated to ROV teleoperation on mobile or desktop computers. In fact, this interface allows sending commands to the ROV (2) as well as supervising sensors' data from the ROV (1, 3 and 5). It also enables a user to use various functionalities (Chat, various Web services, etc.) in the interface (part 4 and 6).

As mentioned, the physical layer (Layer 0) is decomposed into sub-components according to Ellis' 3C model, while providing and managing specific services for communication, coordination and production. Hence, our Web interface shown, in Figure 4, has the following structure:

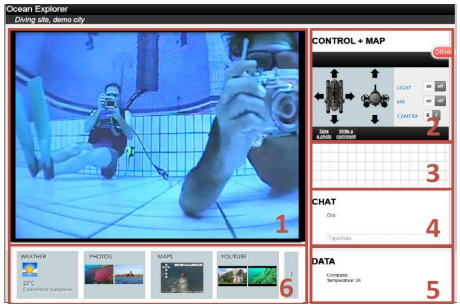


Figure 4: Web HRI for mobile devices and desktop PCs

- **Communication**: The communication space offers to users means to exchange information relative to ROV teleoperation missions. This space, represented by the region (4) of the interface, is based on a textual communication. More communication services can be added, such as audio and video services. However, in this stage of the project, an essential design constraint is to keep the interface as easy-to-use as possible, since we consider that our system will be used by the general public. Adding more functionalities in the interface can be overwhelming for systems' first use.

- **Coordination**: The coordination space (2) implements commands enabling users to create collaborative diving trajectories. It also enables allocating users to various diving paths in teleoperation sessions.

- **Production**: The production space (1) is crucial for visualizing underwater sites from the ROV's camera as video streams. Hence, this space gives users a visual feedback of their diving trajectories that are being executed by the ROV. Also, our system uses AR technologies by the reorganization of 2D real markers, as shown in Figure 5. Those markers can be used to add multimedia data (text, images, videos, fauna and flora 3D models, etc.), as well as to localize the ROV using its camera, which adds a rich interactive experience.

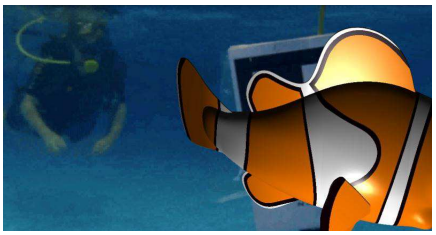

Figure 5: Augmented Reality to display 3D models

**VR HRI**

The second HRI we present in this paper is a VR interface. In fact, in order to effectively teleoperate a remote robot, HRIs must provide tools to perceive the remote environment, to make decisions, and to generate commands. We cite similar work such as in [12, 19] or [14] that introduces a ROV safety domain. Furthermore, we attempt to maximize information transfer while minimizing cognitive and sensorimotor workload. In our system, we used a multimodal interface that contains stereoscopic viewing as well as a haptic feedback for a more intuitive HRI. In fact, it aims to reduce training and overcome the unfamiliarity in using VR systems. This interface improves the feeling of presence and awareness among users doing a virtual seabed exploration. Furthermore, it provides two types of diving scenarios: 1) a simulated dive in a virtual environment without real control of the ROV, which enable users to learn and test a path before doing a real exploration, and 2) a dive in the Mixed Reality (MR) environment via ROV teleoperation.

*VR/AR semi-immersive platform*

The human scale semi-immersive platform used is composed of a large screen (3.2mX2.4m) and a DLP projector (120Hz MIRAG E4000) that provides active stereoscopy. We use stereoscopic glasses (Crystal Eyes 3) and their corresponding transmitter. The interaction is provided by a device having six degrees of freedom force feedback called SPIDAR (SPace Interface Device for Artificial Reality), shown in Figure 6 (1).

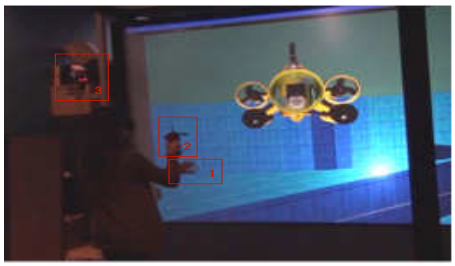


Figure 6: Semi-immersive VR/AR platform

The control system is ensured by a handheld flystick (2) that includes a set of markers for real time tracking using two infrared cameras (3) (ARTTrack1). Communication between the VR/AR platform and the ROV is made via the internet, similarly to the Web HRI.

*VR teleoperation interface*

A virtual environment in our system consists of simulating seabed diving sites, and the navigation task in the virtual scene is done through ROV's teleoperation. Thus, we created a virtual ROV, shown in Figure 7, in order to simulate movements of the real ROV. The navigation using the virtual ROV is done according to the marker placed on the SPIDAR's effector, which reproduces users' position movements. By manipulating the virtual ROV, the operator is actually controlling the real ROV. The system's control allows managing robot's features (camera switch, activate lights) as well as showing instructions for effective usage.

The visual modality is used to display: 1) live video stream from the ROV's camera, means to control the camera as well as a switch from virtual to real teleoperation; 2) virtual and interactive 3D environment representing the explored site; and 3) a top view 2D map of the explored site. Other information, such as sensor data, is also displayed. The force feedback can simulate collisions and increase the feeling of navigation in water (viscosity, marine current, etc.). Furthermore, audible information completes the data offered through the virtual diving interface, which is used

to inform the diver of events occurring, such as the thrusters' sound (indicating the ROV's speed, as well as any troubleshooting issues).

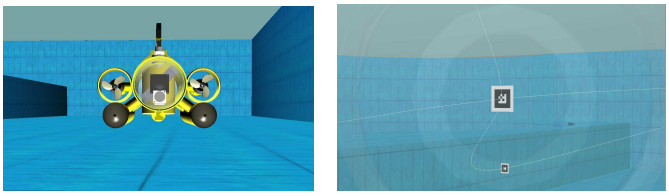

Figure 7: (a) View from a third camera in the virtual test environment (swimming pool), (b) View from the ROV's embedded camera

This multimodality is also used for 3D interaction assistance. In fact, while assembling different issues related to human, environment and teleoperation factors, we observed some technical constraints, such as loss of ROV maneuverability and transmission delays, which affect system's use (navigation precision, robot's safety and divers' spatial awareness). Hence, we applied an assistance model [9] that includes a set of guides (virtual fixtures) in order to remedy these constraints. As an example of a virtual fixture, the operator must choose a diving path that he/she must follow. To be as accurate as possible, we show a 3D curve representing this path as well as an arrow directed towards the trajectory.

In order to assist the user, our system can switch between real and virtual navigation. In fact, the user can choose to control the real ROV through its embedded video camera. In this case, we calculate the corresponding trajectory of the virtual ROV (not being visualized). The user can also control the virtual ROV through the VR interface. In this case, the real ROV will respond to the commands sent to the virtual ROV, and will execute them as well. We also changed system's autonomy through correspondence between the real and virtual ROV. In fact, the coherence between real ROV's position and the virtual one is provided by real 2D markers (shown in Figure 5). When these markers are detected by the real ROV's camera, an estimation of the camera's position is achieved. Once the position (and orientation) of the ROV is calculated according to 2D markers' positions, we calculate the position of the real ROV in its environment. As the positions of the 2D markers are known, we use them as waypoints on the diving path.

## DISCUSSION

The originality of our model is the use of existing software architecture models in order to create an interactive system for collaborative robot teleoperation. Our model is inspired by the Arch and Dewan's models for separating the core functionality (logic of the application) from its interfaces, and thus carrying with it many essential properties such as flexibility, which is crucial in the CSCW domain. Indeed, the two layers constituting the FC are both shared and handle exclusively the services dedicated to robot teleoperation. A functional core adaptor (not discussed in this paper) situated between the functional core and the physical layer handles users' private domain-dependent objects. Furthermore, the functional breakdown according to the 3C model contains several properties. In fact, from the implementation's perspective, it will result in a greater modularity. For example, it would be easier to add a communication service through a video stream mechanism without affecting existing services in the system. This enables the addition of independent and heterogeneous services in order to improve the distribution of features.

In Figure 8, we can see the application of our software architecture for robot teleoperation. Recall that the particularity of our software architecture is the separation of the interfaces from the FC. Hence, our system gives access to various interactive interfaces to order to collaboratively teleoperate the ROV. We have implemented two HRIs that offer a rich interactive and collaborative experience for underwater exploration through ROV teleoperation. In our system, the FC is responsible for offering various teleoperation services, as well as creating an infrastructure through interactive interfaces for issuing commands to the real ROV.

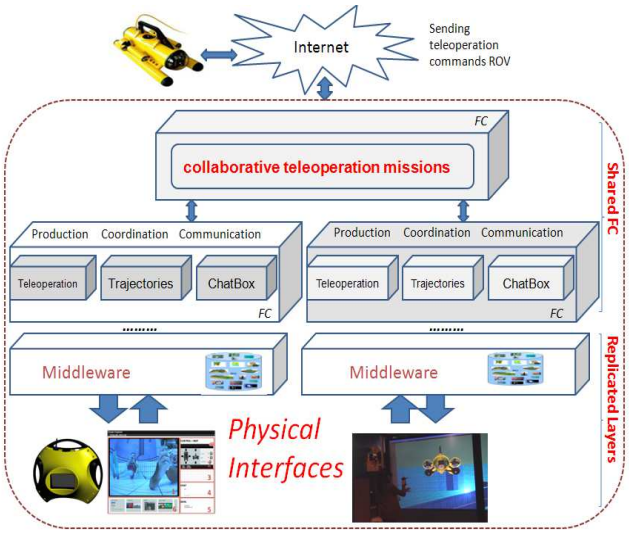


Figure 8: Software architecture applied to ROV teleoperation

## CONCLUSION

In this paper, we have proposed a software architecture for interactive and collaborative teleoperation of an underwater robot using two interactive interfaces. One crucial particularity of our software architecture is the separation of the interfaces from the system's FC, which allows a greater flexibility in teleoperating the robot. Furthermore, we have explicitly taken in consideration the collaborative aspect of the system, in order to allow multiple users to communicate and coordinate for ROV teleoperation. This fact brings a rich interactive and collaborative experience.

However, robot teleoperation over the internet is, sadly, limited in communication possibilities in real time

situations. In fact, it is almost impossible to predict the network path taken by the packets to reach the underwater ROV. Hence, an interaction between users and the distant robot is not guaranteed without any network delays. For this reason, our short-term goal is to add a Quality of Service (QoS) component in the software architecture proposed, which enables handling none-functional attributes such as bandwidth, response time, packet loss, etc. Another objective is to adapt the teleoperation interface on the VR platform according to the 3C model. Indeed, we want to optimize the collaborative experience with other users using the Web interfaces to teleoperate the robot. One possibility is to integrate an audio mechanism (3C model's communication space) in order to enhance the communication process (as the use of keyboards is not possible on the VR platform). Also, we aim at adding a visual feedback of teleoperation trajectories created using the VR platform, in order to diffuse them on the system's Web interfaces. This fact enhances the coordination and cooperation between users using both interfaces (Web and VR) simultaneously. We believe that our system constitutes a first step towards a rich interactive experience for underwater robot teleoperation, both for the general public (through web interfaces), and expert users (VR platform).


## ACKNOWLEDGMENTS

This work is done in the context of the Digital Ocean Europe project funded by the European Commission (FP7).


## REFERENCES


1. Alvarez, B., Iborra, A., Alonso, A. and de la Puente, J.A.Reference architecture for robot teleoperation: development details and practical use. *Journal of Control Engineering Practice*, Volume 9, Number 4, Elsevier, 395-402, 2011
2. Álvarez, B., Ortiz, F., Martínez, A., Sánchez, P., Pastor, J.A., and Iborra. A. Towards a Generic Software Architecture for a service robot controller. *In the 15th Triennial World Congress, Barcelona, Spain*, 2002
3. Bass, L. A metamodel for the runtime architecture of an interactive system, *User Interface Developers' Workshop*, SIGCHI Bull. 24(1) (1992).
4. Bruzzone, G., Bono R., Caccia M., Coletta P., Veruggio G., "Internet-based teleoperation of the Romeo ROV in the arctic region", *Manoeuvring and control of marine craft 2003 (MCMC 2003)*: 6th IFAC Conference, Elsevier Science Ltd, 2004.
5. Calvary, G., Coutaz, J. and Nigay, L. From Single-User Architectural Design to PAC*: a Generic Software Architecture Model for CSCW. *In the ACM Conference on Human Factors and Computing Systems (CHI'97)*, 1997, pages 242-249, ACM Press.
6. Cheaib, N., Otmane, S. and Mallem. Tailorable Groupware Design based on the 3C Model. *In the International Journal of Cooperative Information Systems*. Volume 20, Number 4, 405-439, 2011.
7. Dewan, P. Architectures for collaborative applications, *Journal of Computer Supported Cooperative Work (CSCW), Trends in Software* (John Wiley & Sons, Chichester, 1999), 169–194.
8. Ellis, C.A.Conceptual model for groupware. *Proc of CSCW 1995* (ACM Press, New York), 79-88
9. Essabbah, M., Otmane, S. Hérisson, J and Mallem, M. A New Approach to Design an Interactive System for Molecular Analysis, Lecture Notes in Computer Science (LNCS 5613), Human-Computer Interaction, (HCII 2009), pages 713-722, Springer-Verlag, 2009.
10. Fuks, H., Raposo, A.B., Gerosa, M.A and de Lucena, C.J.P. Applying the 3C model to groupware engineering. *International Journal of Cooperative Information Systems*, v.14, n. 2-3, 299-328, 2005
11. Grange, S., Fong T., Charles Baur C. Effective Vehicle Teleoperation on the World Wide Web. *In the IEEE International Conference on Robotics and Automation (ICRA 2000), San Francisco, CA*, 2007-2012, 2000.
12. Hamzah M. S. M., Zakaria M., Abd Jalil M. F. I., and Zamli K. Z. 3D virtual simulation software for underwater application. *In 2nd International Conference Underwater System Technology*, 2008.
13. Jenkyns R., "NEPTUNE Canada: Data integrity from the seafloor to your (Virtual) Door", *in IEEE OCEANS 2010*, 1-7, 2010.
14. Lin Q, and and Kuo, C. On applying virtual reality to underwater robot teleoperation and pilot training. *In the International Journal of Virtual Reality*, Volume 5, Number 1, 2001.
15. Monferrer, A. and Bonyuet, D. Cooperative robot teleoperation through virtual reality interfaces. *In the Sixth International Conference on Information Visualisation,* 243-248, 2002.
16. Oliveira, F.F., Antunes, J.C.P., and Guizzardi, R.S.S. Towards a collaboration ontology. *Proc. of the Brazilian Workshop on Ontologies and Metamodels for Software and Data Engineering.*
17. Rekleitis, I., Dudek, G., Schoueri, Y, Giguere, P and Sattar J. Telepresence across the Ocean. *In the 2010 Canadian Conference on Computer and Robot Vision,* 261-268, 2010.
18. Roston J., Bradley C., Cooperstock JR. Underwater window: high definition video on VENUS and NEPTUNE, *in IEEE OCEANS 2007*, 1-8, 2007
19. Santamaria, J, Opdenbosch A. Monitoring Underwater Operations with Virtual Environments. *In Offshore Technology Conference, 2002.*
20. Y. Laurillau and L. Nigay. Clover architecture for groupware. In the Proc CSCW, ACM, 236-245, 2002