International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:10, No:9, 2016

Architecture Design of the Robots Operability Assessment Simulation Testbed

Sang Yeong Choi, Woo Sung Park

***Abstract*—**This paper presents the architecture design of the robot operability assessment simulation testbed (called "ROAST") for the resolution of robot operability problems occurred during interactions between human operators and robots. The basic idea of the ROAST architecture design is to enable the easy composition of legacy or new simulation models according to its purpose. ROAST architecture is based on IEEE1516 High Level Architecture (HLA) of defense modeling and simulation. The ROAST architecture is expected to provide the foundation framework for the easy construction of a simulation testbed to order to assess the robot operability during the robotic system design. Some of ROAST implementations and its usefulness are demonstrated through a simple illustrative example.

***Keywords*—**Robotic system, modeling and simulation, Simulation architecture.

I. Introduction

IN recent, robotic systems are increasingly applied to the various domains such as medical science, search and rescue operations, and military defense. Although most of the modern robots are designed to a high degree of autonomy, a human may not be completely excluded from the robot operations. A human operator will still intervene in the robot operation and tele-operate them to achieve his or her mission. Semi-autonomous robot, for an example, at first may be moving and performing tasks without human attentions. A human operator will be keeping the situation awareness through user interfaces. In a certain condition where robots encounter obstacles or hostiles, the robots may require human interventions. The intervention may be a simple corrective command or a full-scale tele-operation. Then, the robot control may be ceded to the human operator and may be ceded back to the robots. During such interactions between the human operator and robots, factors such as human workload, human error, situation awareness, control hand-over policy, communication delay, level of autonomy, etc. may influence the performance of robot mission. We call such issues "Robots Operability" (RO) problems.

The RO problems may be addressed by the physical experimentations in the field laboratory. The typical example cases are in [1], but such a field laboratory costs a lot of time and effort. Alternatively, simulation models can benefit if its fidelity is sufficiently enough to the purpose of experimentations. Because the simulation models are less costly to construct, further much easier to control the experiment variables and conditions [2]-[6]. However, the current breed of the RO simulation models lacks reusability and interoperability.

This paper presents the architecture design of the ROAST, on which we can construct the simulation models by easy and cheap composition of legacy or new simulation models in a reusable way. The ROAST architecture adopted IEEE1516 HLA of defense modeling and simulation [7], because HLA is widely accepted towards such ends.

This paper is organized as follows. In Section II, the theoretical backgrounds of the robot operability problems will be briefed. Section III describes our use cases of the ROAST. Section IV explains the ROAST architecture. In Section V, we will show some implementations of ROAST, and an illustrative example of the test cases. In the last chapter, we conclude with a description of further works.

II. Theoretical Backgrounds

Robotic systems are similar to the joint cognitive systems. Hollnagel and Woods [8] introduced a joint cognitive system (JCS), which is a system that can modify its behavior on the basis of past experience so as to achieve specific anti-entropic ends. JCS can be any combination of humans and machines (technological artifacts) or humans and humans (social groups/ organizations). We believe Hollnagel's JCS can be a good starting point for our problems. Thus, firstly we took the JCS as a theoretical foundation underlying operability problems of our robots, and we extended it to fit for our purpose, which is shown in Fig. 1.

We have four tiers in Fig. 1: operator, interface, robots, and mission objective. The operator layer deals with the human cognitive process such as perception of situation, evaluation of goal, selection of course of action (COA), and implementation of COA. The interface layer is for information and message visualization to help the operator's situation awareness and for the expression of command and control of remote robots. Further cognitive tasks are performed on this interface layer by a human operator. The robot layer deals with robot functional autonomy according to its automation level. The final layer is for mission objective. The mission objective has to be achieved by the human-robot interaction.

Sang Yeong Choi is with MyongJi University, Cheoin-gu, Gyeonggi-do, Korea (e-mail: metayoung@ gmail.com).
Woo Sung Park is with the MyongJi University, Cheoin-gu, Gyeonggi-do, Korea.

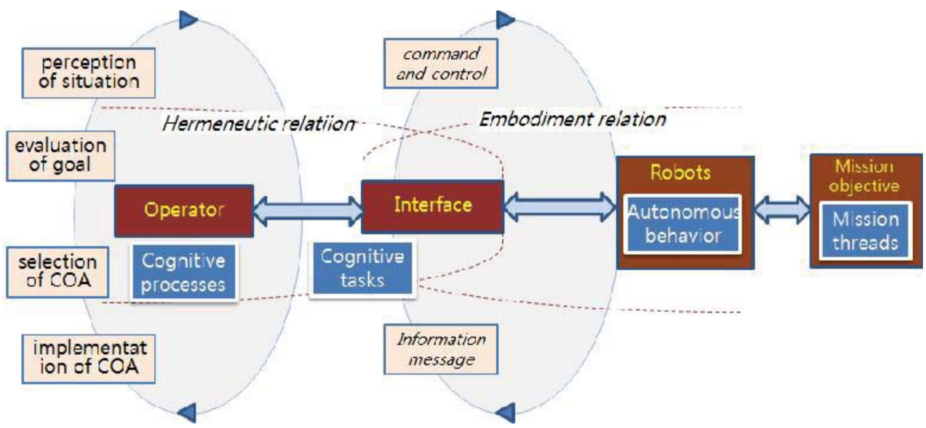

Fig. 1 Extension of Hollnagel's JCS

III. Use Cases of ROAST

The primary purpose of the ROAST is to evaluate robot management (for example, human operator/team workload, task diffusion, fan-out, etc.), user interface, robot autonomy, and its mission effectiveness during the conceptual design of robotic system. Towards this ends, major common use cases of ROAST have been identified as follows.

- *UC1: Evaluate cognitive cask performance*
- *UC1.1: Simulate control center UI configuration*
- *UC1.2: Simulate human operator*
- *UC2: Evaluate human-robot interaction*
- *UC2.1: Simulate robot autonomous behavior*
- *UC2.2: Simulate information message exchange*
- *UC3: Evaluate mission thread*
- *UC3.1: Simulate goals and obstacles*
- *UC3.2: Simulate mission plan*
- *UC3.3: Simulate environment*

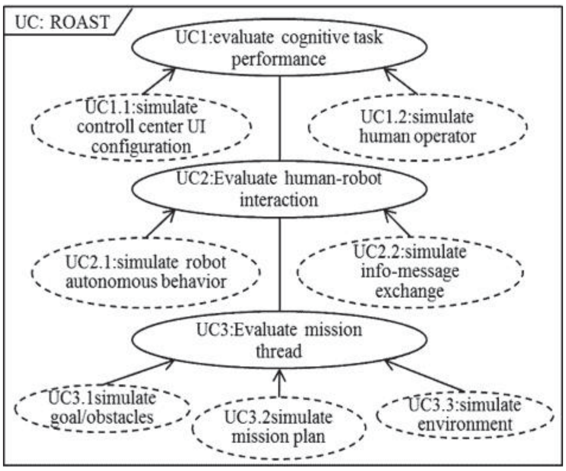

Fig. 2 Use Cases for the ROAST

UC1 (evaluate cognitive task performance) is to evaluate the operator's cognitive task performance in a mission scenario; e.g., perception, evaluating the goal, selecting COA, implementing COA [9]. *UC1.1 (simulate control center UI configuration)* and *UC1.2 (simulate human operator)* are ROAST functionalities to support the UC1.

UC 2(evaluate human-robot interaction) is to evaluate the human-robot interaction. For example, fan-out [10], task saturation and task diffusion [11], communication effects, and so on. *UC2.1 (simulate robot autonomous behavior)* and *UC2.2 (simulate information message exchange)* are the ROAST functionalities to support the UC2.

UC3 (evaluate mission thread) is to evaluate mission thread. For example, search and rescue, mine sweeping, bomb sniffer, etc. *UC3.1 (simulate goals and obstacles)*, *UC3.2 (simulate mission plan)*, and *UC3.3 (simulate environment)* are the ROAST functionalities to support the UC3. The ROAST functionalities may be implemented by such object classes for examples Operator, UI (User Interface), Robot System, Mission System. The use cases are shown schematically in the use case diagram in Fig. 2.

IV. ROAST Architecture

ROAST architecture was designed to provide the foundation framework for the easy construction of a simulation testbed in order to evaluate the robot operations in the robotic system design. We adopted IEEE11516 HLA as the foundation of ROAST Architecture, because the ROAST needs to be formed as a distributed simulation with reusability and interoperability ensuring a high fidelity of robot operability simulation. IEEE11516 HLA is widely accepted to this ends.

Fig. 3 shows ROAST architecture in a schematic diagram. In the ROAST architecture, ROAST federation can have constituent models, called "ROAST federate". The ROAST federate candidates are SimulationController, ControlCenter, RobotSystem, MissionSystem. The ROAST federate candidates contribute to a ROAST federation execution according to the operability evaluation plan by joining itself to the ROAST federation.

SimulationController federate is to represent the simulation controller for start, stop, end, logging, statistical analysis. ControlCenter federate is to represent human operators or teams associated *UC1.1: simulate control center UI configuration*, and the user interface of control center associated with *UC1.2: simulate human operator*.

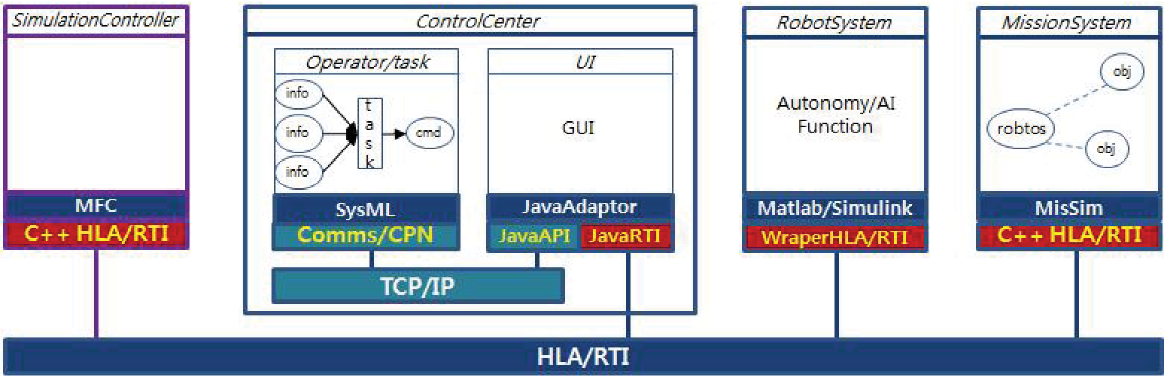

Fig. 3 ROAST Architecture

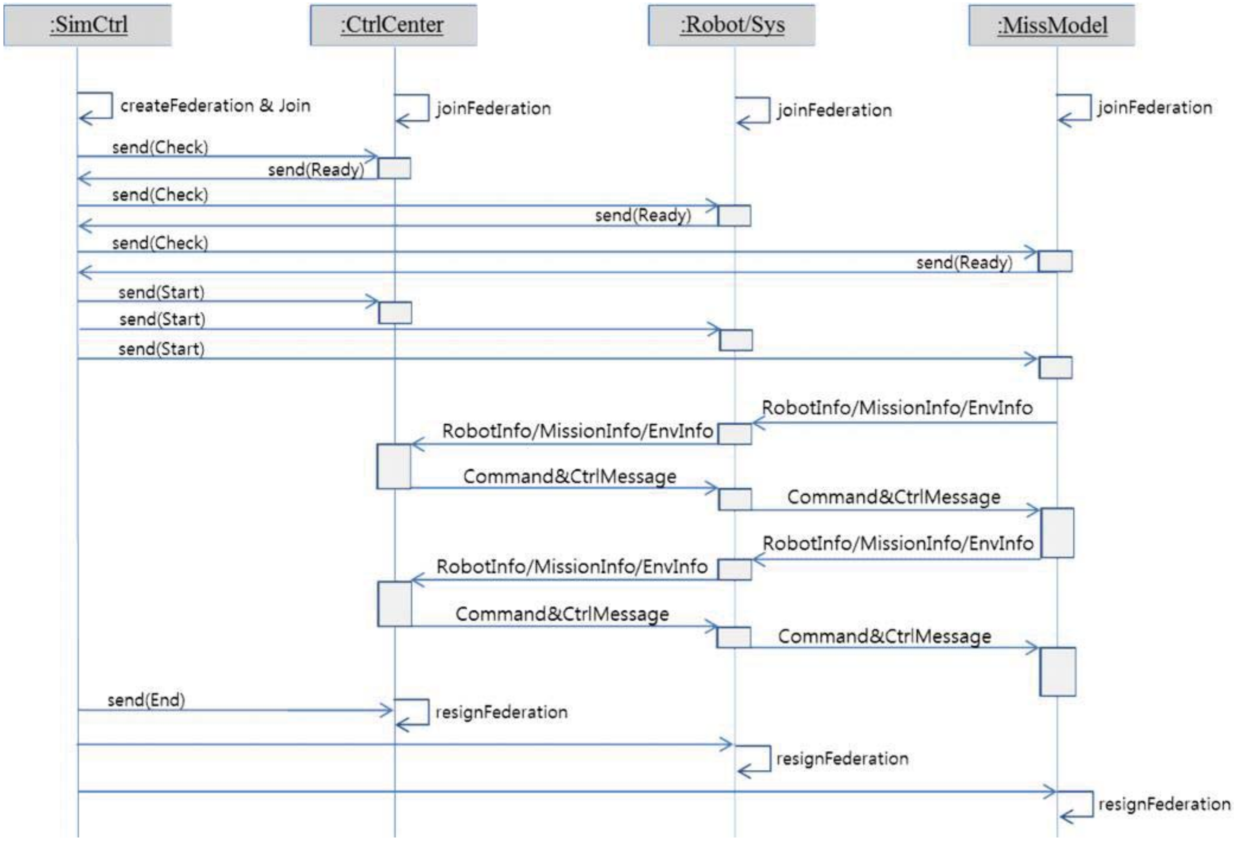

Fig. 4 ROAST federation scenario for *UC2.2(simulate information message exchange)*

RobotSystem federate is to represent the robot autonomous behavior associated with *UC2.1: simulate robot autonomous behavior* and information-exchange communication associated with *UC2.2: simulate information message exchange*.

MissionSystem federate represents the goals and obstacle associated with *UC3.1: simulate goals and obstacles*, mission plan associated with *UC3.2: simulate mission plan*, and environment associated with *UC3.3: simulate environment*

For the ROAST federation execution, SimulationController federate and MissionSystem federate are using HLA/RTI services C++ to join the ROAST federation. But, the ControlCenter federate has a Java adaptor for joining the ROAST federation execution, because the ControlCenter federate incorporates the CPN tool which is useful for the human operator task model [12], [13], and can connect with the reusable interface Comms/CPN [14], [15]. Further RobotSystem federate can plug-in the ROAST federation execution by using MATLABHLA-Toolbox service [16], when it is implemented with MATLAB/Simulink tool.

Thus, the ROAST architecture can provide the framework wherein the ROAST federate implements the use cases scenario described in Fig. 2 by joining the ROAST federation execution. For example, the execution scenario for the *UC2.2(simulate information message exchange)* is conceptually described in Fig. 4. This sequence diagram makes a story of the use case *UC2.2* during the ROAST federation execution.

V. Some Implementation and an Illustrative Example

We implemented the ControlCenter federate to be compliant with the ROAST architecture shown in Fig. 2. The ControlCenter embodied the human operator's cognitive task model and user interface model with a simple scenario of the surveillance and reconnaissance mission, where a single operator controls a single robot in a supervisory way. The robot is an unmanned ground vehicle (UGV). The cognitive task model incorporates four major cognitive tasks: control mission plan, control the robot mission execution, control a stuck situation, control the robot movement. These are coded with CPN tool [17] and Java.

We created a ROAST federation execution using MÄK RTI [18]. Then, ControlCenter federate had successfully joined to the ROAST federation execution. Further, we performed the experimental assessment of the human operator's workload by changing the mean time of the requested intervention events in the mission scenario. Fig. 5 shows some results from the experimentation. In these results, a human workload (visual, auditory, cognitive, psychomotor) increases as the mean time between the requested intervention events decreases to 25sec (case1), 20sec (case2), 15sec (case3), 10sec (case4), 5sec (case5) with an exponential distribution, as we may expect. It is noted that the maximum level of workload for a single operator is 7 [19]. The intervention event occurrence time can be related to a concept of interaction effort, which is one of common metrics used in the human-robot interaction [20].

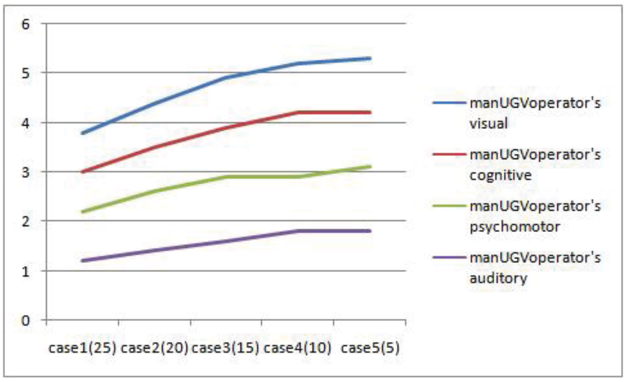

Fig. 5 Test case for the rescue mission operation

VI. Conclusions and Further Works

We presented the architecture of the ROAST for the robot operability simulation, which is based on IEEE1516 HLA promoted for the reusability and interoperability of the defense simulation components. In conclusions, our first trial was successful to implement a ROAST federate compliant with the ROAST architecture and performed a test case assessment of a human operator's workload given a surveillance and reconnaissance mission scenario. Through this test case, it is ensured that the ROAST federation architecture may provide the foundation framework for the easy construction of the robot operability simulation. Fuller implementation of ROAST federates is still on going. In the near future, we hope to come up with a complete suite of the ROAST federation solution.

Acknowledgment

This work was supported by a research program (The Specialized Research Center for Future Ground System Analysis) funded by the Agency of Defense Development of Korea and we appreciate it.

References

[1] Michael Lewis, et.al., USARSim: Simulation for the Study of Human-Robot Interaction, Journal of Cognitive Engineering and Decision Making, Volume 1, Number 1, Spring 2007, pp. 98–120.
[2] MIT, RESCHU (Research Environment for Supervisory Control of Heterogeneous Unmanned vehicles).
[3] Naval Postgraduate School, RESCHU-SA (Research Environment for Supervisory Control of Heterogeneous Unmanned vehicles –Swarm Attack).
[4] Naval Postgraduate School MOVES, SAWSI (Semi-Autonomous Wingman Supervisory Interface).
[5] Connecticut University, MAVIES (Multi-Autonomous Vehicle Insertion Extraction System).
[6] Science Application International, CART (Combat Automation Requirements Testbed).
[7] IEEE 1516–2010 – Standard for Modeling and Simulation High Level Architecture – Framework and Rules.
[8] Hollnagel, E. & Woods, D.D. (2005) Joint Cognitive Systems. Foundations of Cognitive Systems Engineering. Boca Raton, FL: CRC Press, Taylor & Francis Group.
[9] Wickens et al. 2013. Engineering Psychology and Human Performance. New York: Pearson Education, Inc., pp.3~6.
[10] D. R. Olsen and S. B. Wood, “"Fan-out: Measuring human control of multiple robots,”" in Proceedings of the SIGCHI conference on Human factors in computing systems. Vienna, Austria: ACM, 2004, pp. 231–238.
[11] Whetten, J., Goodrich, M., Guo, Y.: Beyond Robot Fan-Out: Towards Multi-Operator Supervisory Control, In: IEEE International Conference on Systems Man and Cybernetics(SMC) pp.2008-2015(2010)
[12] Kurt Jensen (Ed.), Tenth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools Aarhus, Denmark, October 19-21, 2009.
[13] Tom Kontogiannis, Integration of task networks and cognitive user models using coloured Petri nets and its application to job design for safety and productivity, Cogn Tech Work (2005) 7: 241–261.
[14] G. Gallasch and L. M. Kristensen. COMMS/CPN: A communication infrastructure for external communication with Design/CPN. In K. Jensen, editor, Third Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, DAIMI PB-554, pp. 75–91.
[15] Himanshu Neema, et.al., Rapid Synthesis of Multi-Model Simulations for computation Experiments in C2, GMU-AFCEA SYMPOSIUM on Critical Issues in C4I Lansdowne, VA May 19, 2009.
[16] HLA Toolbox ™ The MATLAB® interface to HLA, http://www.forwardsim.com.
[17] Kurt Jensen, Coloured Petri Nets, Springer, 2009.
[18] MÄK RTI: HLA Run Time Infrastructure, http://www.mak.com.
[19] McCrasken, J. H. and Aldrich, T. B. 1984. “Analysis of selected LHX mission functions: Implications for operator workload and system automation goals.” Technical Note ASI479-024-84. Fort Rucker, AL: Army Research Institute Aviation Research and Development Activity.
[20] D. R. Olsen and M. A. Goodrich, "Metrics for Evaluating Human-Robot Interactions," presented at Performance Metrics for Intelligent Systems, Gaithersburg, MD, 2003.

Sang Yeong Choi was born in Busan, Korea and went to Korean Military Academy, Korean National Defense University, Cranfield University of UK, where he studied defense science and obtained Ph.D. in 1989. He is a research professor in The Specialized Research Center for Future Ground System Analysis, MyongJi University. Previously, he was a professor at Korean National Defense University.

Woo Sung Park was born in Seoul, Korea and went to MyongJi University, Korea, where he studied mechanical engineering and obtained MSc in 2013. He is a researcher in The Specialized Research Center for Future Ground System Analysis, MyongJi University.