DOI: 10.2507/27th.daaam.proceedings.016

PROGRAMMING OF ROBOT SYNERGISM IN MULTI-AGENT SIMULATORS

Valentin Pryanichnikov, Denis Davydov, Kirill Kirsanov & Stanislav Eprikov

This Publication has to be referred as: Pryanichnikov, V[alentin] E[.]; Davydov, D[enis]; Kirsanov, K[irill] & Eprikov, S[tanislav] (2016). Programming of Robot Synergism in Multi-Agent Simulators, Proceedings of the 27th DAAAM International Symposium, pp.0109-0115, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-08-2, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/27th.daaam.proceedings.016

Abstract
This article reviews the problem of building software systems for control of distributed hardware sets of mechatronic systems, mobile robots and their virtual models. For this purpose, application-dependent software ("middleware") was built, which allows performing operations on asynchronous data buses with correct network input/output of various controlled devices along with combination of work of real robots and their virtual images. One of the well-known approaches to building adequate mathematic models of mobile robots is the use of V-REP robotic simulator, its abilities for creating new solid models out of the known components and modules. However, such an approach allows us to solve the design problem of separately functioning robotic systems. This study describes an approach to create a distributed robotic simulator, which allows programming the control of mobile service robots by generating the so-called "synergism" – generic templates of more elementary (or previously constructed) actions.

Keywords: mobile service robots with distributed control; robotic simulator; simulated environment; V-REP.

1. Introduction

On the basis of the conducted research of various options of technical implementation of the project for creating a network of Internet laboratories called "Intellectlnaya Robotronika" and building supervisory control via Internet by mobile training robots AMUR (*AMYP*) (International Laboratory "Sensorika") and Robotino (Festo) with organization of video monitoring of command execution, a new program architecture has been proposed for the distributed data measuring and control system (DMCS). This approach to building DMCS has shown stability against network delays and allowed to overcome the main disadvantages of the existing approaches by means of constructing three principal components (Fig.1):

- Worker – This component implements a parallel process, which performs communication between the mechatronic device and other system components.
- Connector – This component implements a network protocol for access to mechatronic components.
- Locator / Combinator – This component enables the search of mechatronic components within the network, accumulates, and provides access to, data buses, provides dispatching of messages and aggregation service.

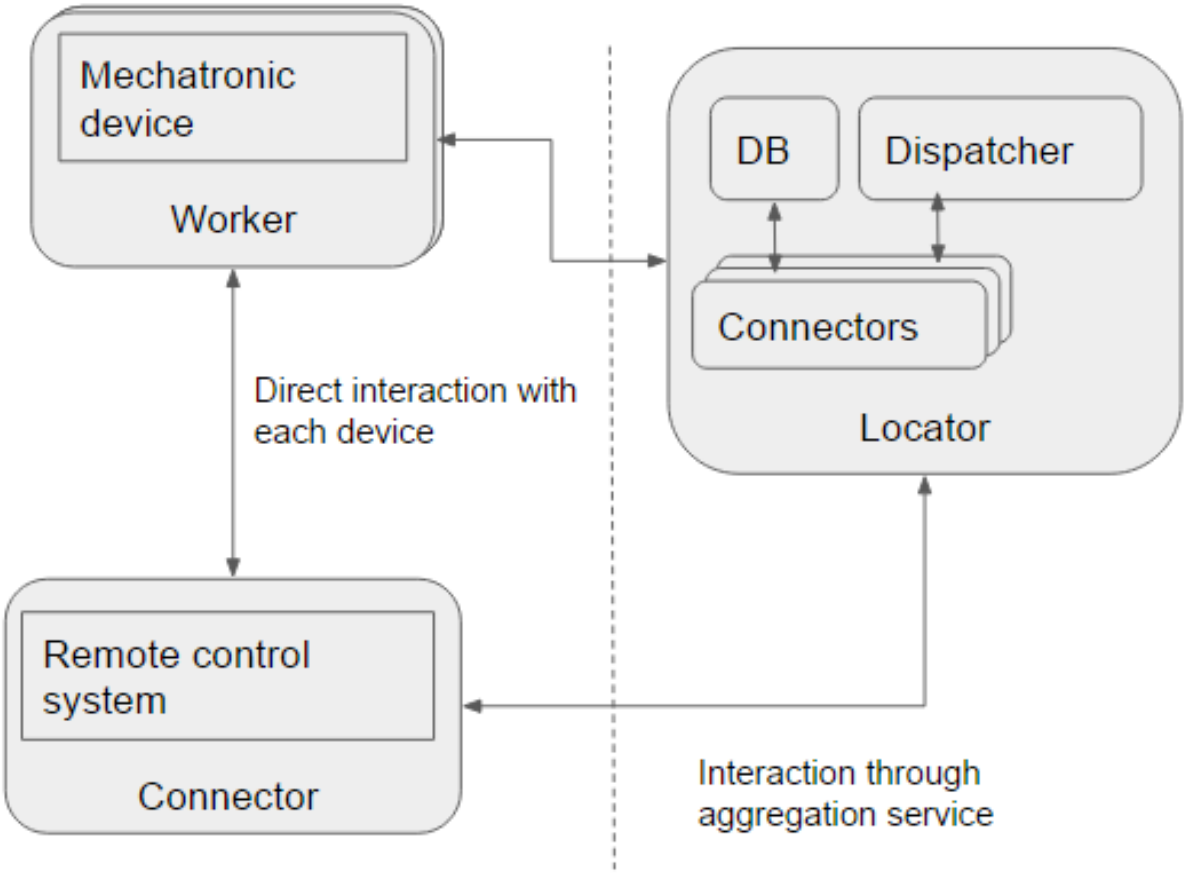

Fig. 1. The scheme of interaction of software components: Onboard devices interact directly with each other. Remote control system gains access to aggregating service. It allows for direct communication with remote control system for debugging purposes

In consequence of adding a database into the structure of controlling software, there appears an opportunity to analyze and compare the measurements recorded and commands either in real time mode, or after the event. DB data are stored in "rough form", i.e. without additional conversion, and for this reason one needs to know their structure in order to work with them.

This information is contained in a manifest file, which is a part of the driver. It contains information on the transmitted data, commands and their SI-system values as well as on operation frequencies of the main loops. This metadata allows organizing the interoperation of mechatronic devices on a higher level by means of intelligent interpolation and extrapolation, synchronization of loops with different frequencies.

Also, the use of DB makes it possible to simplify the indication process of key synergies of the mechatronic system (Fig.2). For this task, a prototype of the editing tool having access to both reference measurements and their interpolated/extrapolated forms.

It provides the possibility to:

1. View the measurements and commands at various scales;
2. Automatically calculate the basic statistic values – mathematical expectation, dispersion, correlation with other measurements or commands;
3. Provide displacement and scaling of measurements or commands in order to model a new behavior of the mechatronic system. However, this requires certain models to be implemented, for example, with the help of virtual simulators or similar software tools called "physics engines". These programs often reflect the real mechanical processes, are not based on dynamic equations for mechanical systems, i.e. they are phenomenological models.

2. The use of V-Rep simulator for robotic systems modeling

In the tasks of robotic systems design, it's often necessary to preliminarily test the algorithms for control and construction of the robot's virtual model. Today, there are a set of simulation environments is available for these purposes, for example, Gazebo, Microsoft Robotics Developer Studio as well as V-REP.

Fig. 2. An example of ribbon cutting synergism by Robotino robot's manipulator

V-REP is a simulation environment for various types of robots, at that, the user does not have to have physical access to the real robot, which saves time and costs. Its most distinctive features are: its free-of-charge basis for non-commercial use, various methods of the simulated robot programming and a vast range of robot types, which can be modeled simultaneously.

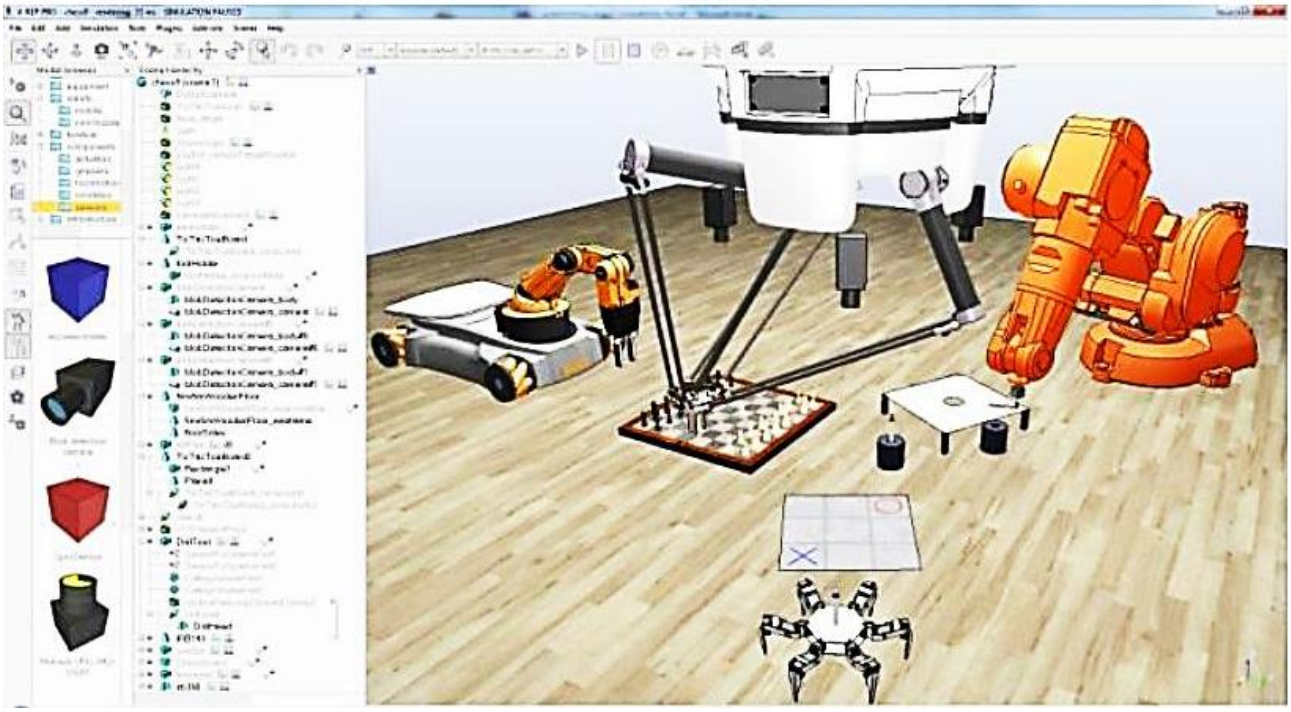

Fig.3. An example of simulation cycle in V-REP simulator

V-REP environment provides a convenient interface for visualization of the robot's actions in 3D virtual space mycg earlier than the real robot prototype will be created (Fig. 3-4).
V-REP is an ideal tool for: fast prototyping and verification; remote monitoring; quick algorithm development; education robotics and factory automation modeling systems.

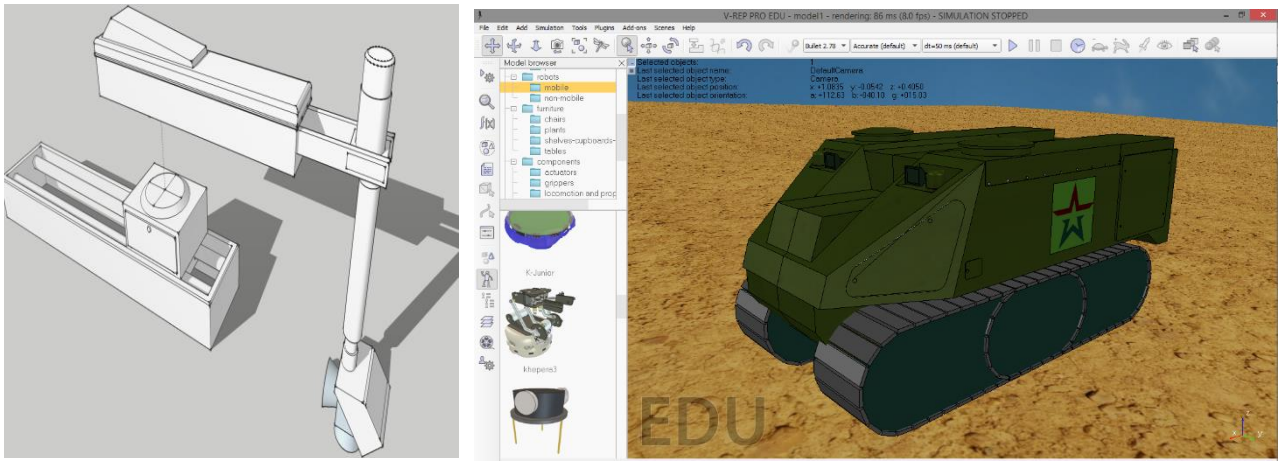

Fig. 4. Building new objects in V-REP simulator

In the simulator, there are already prepared models of such industrial and mobile robots as Kuka LBR4, ABB IRB 140, Pioneer p3dx, etc. On top of that, there is a possibility for a user to create his/her own models out of the existing components.

These components include:

1. Connections. This component may interlink several objects and work in rotation / sliding mode as an engine/actuator.
2. Contactless sensors. They calculate the distances to figures (objects), which are within the range of sensor's detection.
3. Visual sensors. These sensors allow extracting complex images from the simulation scene and work as cameras.
4. Force sensors. They allow writing the applied forces and rotation moments into files and can break if the preset threshold is exceeded.
5. Molds. They are used to simulate the volumes of solid bodies.
6. Rotary tools. They are used for simulation of rotary cutting, laser-beam cutting, etc.
7. Charts. They are able to perform recording of various data, which may vary during the simulation process. The saved data can be displayed on the chart or written to a file.

The simulator supports several programming methods, for example, inline scripts. With this method, it is possible to write a control program right in the simulator without installing any external software and explicit compilation. Scripts are written in a built-in editing program in Lua script language. A script can be opened within or out of stream implementation.

Remote clients. Remote interface provides an opportunity to interact with scene components by means of sockets. On the client side, any software environment can be applied if it has the option to connect dynamic-link libraries. The simulator comes with libraries for remote operation of V-REP for some of the most popular programming languages (C/C++, Python, Java, Matlab, Octave, Lua, Urbi). The advantages of such method are the use of familiar programming environment and integration with the existing solutions.

Despite different approaches to programming of the simulated object, the software interface is almost identical for both client and server.

5. Development of network-type robotic simulator

The disadvantage of V-REP simulator reviewed above is the set of limitation for its use in networks, especially in case of parallel operation of several operators with one robot.

In V-REP simulator, control is performed by means of connection to particular components of the simulated robot. In case of multiple connections, the simulator doesn't perform user authorization and regularization of components available to a user based on privileges. Because of that, situations may arise when the developer gets access to the components of someone else's robot.

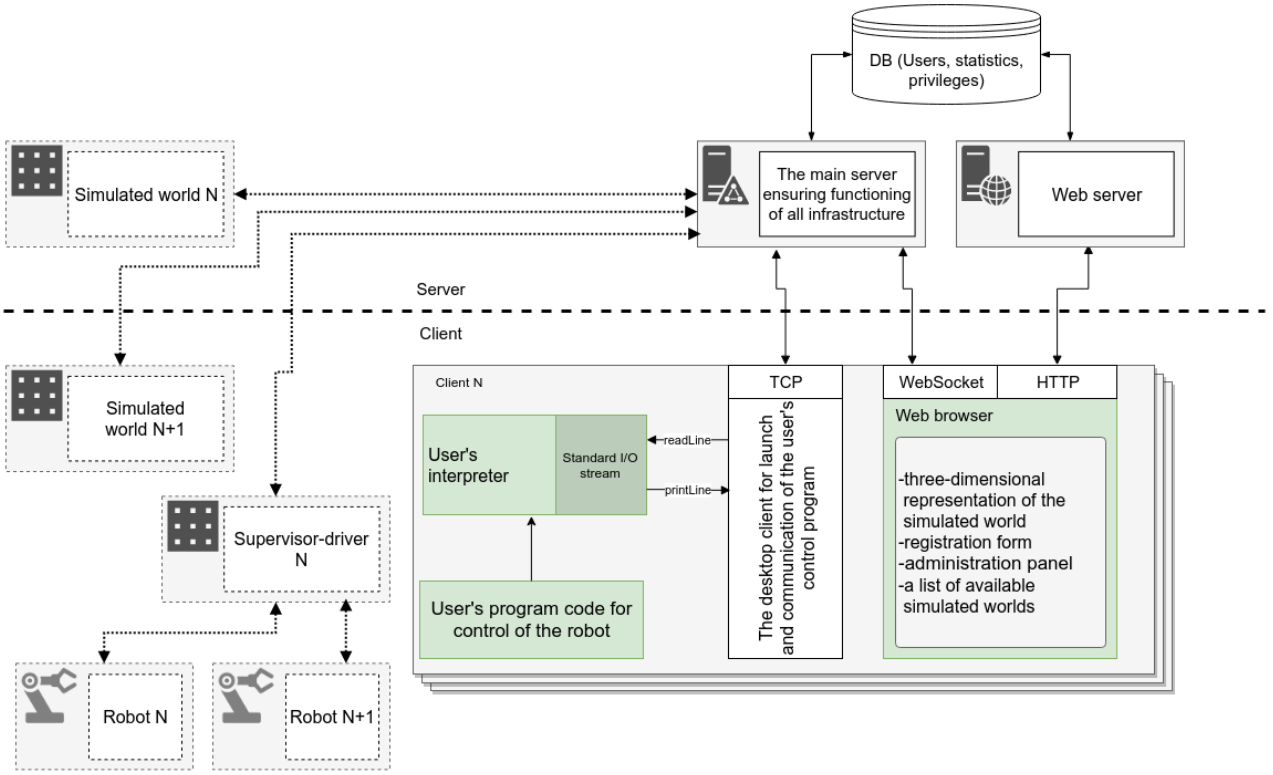

Fig. 5. The structure of the developed network-type robotic simulator

Besides robot programming, it is necessary to set the behavior of the scene's surrounding objects. Such behavior is set by the scene creator and must not be available to be changed by the user. For this purpose, V-REP supports inline scripts in a special scripting language, but it is limited by the simulator's program interface and doesn't allow changing low-level settings of the physics engine.

Cumulatively, the above-named reasons make it more difficult to use V-REP simulator in a multi-user environment. In order to solve these problems, a prototype was developed.

The developed simulator prototype is going to be used for simulation of one or more robots, which can perform competitive or cooperative tasks. A control program can be developed separately for each robot and started up on separate machines.

The simulator's architecture, which has actor model as the basis, allows to scale each component of the system on physically separate computers, and also to design the system with account of high fault tolerance.

Main advantages of the proposed architecture:

- Distributed server. Simulated world (physics engine plus a set of rules) can be started on the client side, which reduces the load on central server;
- Robot control client program works on the client machine, which allows the client to use any software environment, any programming languages, libraries, etc. This reduces the load on central server.
- Robot client control program communicates with the robot by means of standard input/output stream, i.e. only 2 standard functions (printLine, readLine) are needed, which can be found in any programming language. This reduces the simulator entry threshold, allows starting to write the control program immediately without configuring the environment and libraries. Since the system is centralized (started up on the central server), this allows:
    - several robots to take part in one simulated world;
    - gathering the statistics on tasks performed;
    - maintaining the administration of simulated worlds.
- Visual feedback for the developer takes place in a web-browser. In order to work with the simulator, it is only required to download a special launcher program for starting of controlling software.

Such an approach also has a number of disadvantages:

- Security problems. Physics engine works on the client side, and it is potentially vulnerable to external intervention from the client side;

- Client software works on a client, and therefore it becomes possible to receive information from external sources, for example, through manual control of the robot;
- If the physics engine is started on the client, the values pass through the central before they get to the console, and only then the client receives them. This creates or may create a significant delay in feedback for controlling software.

Methods of controlling the disadvantages specified above make it more difficult to use the simulator in real time mode:

- Naturally, the increase in delay between discrete control steps to ~5 seconds when using visual feedback won't allow an opportunity for a person to efficiently control the robot manually, but leaves a possibility to analyze the control afterwards.
- The use of discrete modeling (lockstep model). In this case, all clients are polled and the answer from all clients is expected (with a certain limited time-out). Only after that, the results of using the implemented robot control will be calculated with the use of the physics engine. After all calculations within the system, simulator switches to the next discrete step in time. This allows avoid synchronization of the robot's world model.

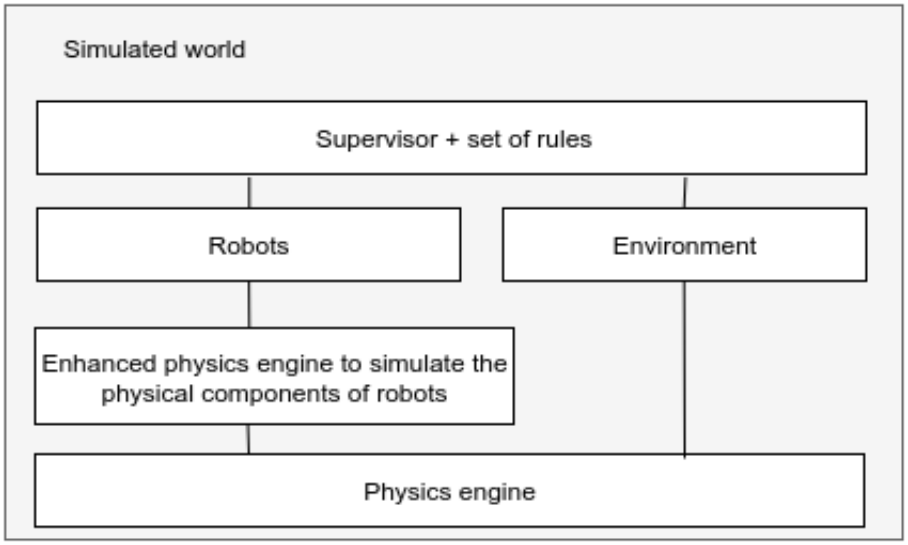

Fig. 6. The structure of the simulated world

In the current implementation of the simulated world, there's no possibility to configure the scene environment and its behaviors through the graphical interface for a user having no programming skills. Any changes in the scene require recompilation of the world. Because of that, V-REP simulator can serve as the simulated world, since it provides simpler scene setting and robot construction processes for people who are not qualified as software developers.

The developed simulator can partially replaces the functionality of other simulators and can be used as an add-in, which expands the functionality of the existing solutions allowing them to operate together. Since any robot control is performed through a centralized server, the client gets a uniform robot control interface. This is achieved by means of manifest files.

The protocol for data transmission between the client and server is generated on the basis of manifest files. They define the robot's structure, its components and possible operations on them. Such robot description in a formal language allows for automatic generation of software interfaces for different programming languages and software environments. Also, such manifests play the role of documentation, which the client-developer can use to review the methods of working with one or another robot. As the result, the user gets a generated software interface, which is connected to the interpreter/compiler. The user can start to work with the robot immediately without thinking about the details of implementation and data transmission method.

6. Conclusion

The developed application-dependent software («middleware») has allowed building a system, which makes it possible to combine the work of real robots and their virtual mathematical models. The main distinctive feature of the developed tools is operation in network access mode to all subsystems of the developed simulator and to real mobile robots (Amur, Robotino) as well. The created technology has allowed programing and reprogramming the mobile robots by generating synergies in network access and even during their functioning. It also allows the use of virtual models.

7. References

[1] Emmerich, W. The impact of research on the development of middleware technology / Wolfgang Emmerich, Mikio Aoyama, Joe Sventek // ACM Transactions on Software Engineering and Methodology. – N. Y.: ACM – 2008. Vol.17, № 4. – pp.19-48.

[2] Pryanichnikov V., Andreev V. The Application of Network Technologies to Constructing Group Controlled Systems with Machine Vision for Mobile Robots // In Proceedings of the 23th DAAAM Intern. Symp. "Intelligent Manufacturing & Automation" 24-27th October 2012 Zadar, Croatia, 2012. – V.23, No.1, – pp.1167 – 1174.

[3] Andreev V.P. Technology Supervisory Control for Mechatronic Devices via the Internet / V.P.Andreev, K.B.Kirsanov, P.F.Pletenev, Yu.V.Poduraev, V.E.Pryanichnikov, E.A.Prysev // 25th DAAAM Intern. Symp. "Intelligent Manufacturing & Automation". Procedia Engineering, 2015. – V.100, – pp.33 – 40.

[4] Andreev V. Education on the basis of virtual learning robotics laboratory and group-controlled robots / Andreev V., Pryanichnikov V., Poduraev Y., Kuvshinov S. // 24th DAAAM Intern. Symp. on Intelligent Manufacturing and Automation, DAAAM 2013. Procedia Engineering, 2014. – V.69. – pp.35 – 40.

[5] Carl Hewitt, et al. Actor Induction and Meta-evaluation // Conference Record of ACM Symposium on Principles of Programming Languages, January 1974.

[6] V-REP simulator documentation URL [electronic resource]. – Access mode: http://www.coppeliarobotics.com/helpFiles/

[7] Rohmer Eric. Singh and Marc Freese. V-REP: a Versatile and Scalable Robot Simulation Framework / Eric Rohmer, P.N. Surya. – 2013.

[8] Pryanichnikov V.E., Andreev V.P., Prysev E.A. Group control of mobile robots, based on the net-technologies // Proceedings of the Int. conference on Robotics for Security & Space (Planet & Earth rovers). St. Peterburg: Poliectnika service. 2010. P. 279–283.

[9] Katalinic B., Pryanichnikov V.E., Ueda K., Kukushkin I., Cesarec P., Kettler R. Bionic Assembly System: hybrid control structure, working scenario and scheduling // Proceedings of 9th National Congress on Theoretical and Applied Mechanics, Brussels. 2012. P. 111–118.

[10] Pyanichnikov V.E. Algorithmic base for remote sensors of mobile robots // Mechatronics, Automation, Control. 2008. № 10(91). P. 10−21.

[11] Kirsanov K., Levinsky B., Pryanichnikov V. Integrating software for intelligent robots // Informational-Measuring and Control Systems, Radiotechnika. 2009. V. 7. № 6. P. 35−43.

[12] Kirillchenko A.A., Pryanichnikov V.E., Rogozin K.V. Limits of validity and reliability of proofs. Scepticism in mathematics, functions, and traditions // Information-Measuring and Control Systems. 2013. V. 11. № 4. P. 57–65.