# Cloud service for development of virtual interactive environments

Valeriya Gribova
Intelligent System Laboratory, Institute of Automation and Control Processes, Russian Academy of Sciences, School of Computer Science, Innovations and Business Systems, Vladivostok State University Economics and Service
Vladivostok, Russia
gribova@iacp.dvo.ru

Leonid Fedorischev
Intelligent System Laboratory, Institute of Automation and Control Processes, Far Eastern Branch of Russian Academy of Sciences
Vladivostok, Russia
fleo1987@mail.ru

***Abstract*—The article presents main ideas, the architecture, and information and software components of the software system for development and maintenance of virtual interactive environments. The aim of the new approach is to simplify the process of creating this kind of software and to involve in the process domain experts and designers**

***Keywords- virtual reality, ontology, cloud computing, agents, visualization***

## I. INTRODUCTION

Extensive use of informational and computer technologies results in advent and evolution of lots of innovative multimedia tools. One of such tools is virtual interactive environments. User can interact actively with environment elements changing them. Virtual interactive environments are widely used in education, medicine, scientific visualization, architecture, and other domains, so the need for their further development is beyond any doubt.

There are a lot of tools, special software packages for creating virtual interactive environments, among them are Delphin, ToolBook, Virtools, CAVE, WorldToolKit, Avango, Lightning, Juggler, Unity3D, Virtools, Alternativa3D, and others. However, development and maintenance of virtual interactive environments are still a complicated process. There are a lot of different factors, main of them are: the development involves a process of coding complicated scripts or programs using imperative programming languages ) followed by their integration into a working system; domain experts can't directly (without engineers and programmers) take part in the development of such applications because the experts don't have special, understandable for them, tools of development and maintenance. As a result, the life cycle of such kind of software is still very short. Modification and extension of functional features of them are so complicated processes [1-4].

The aim of this paper is to describe a software system for development and maintenance of virtual interactive environments simplifying these processes.

## II. MAIN IDEAS OF THE SOFTWARE SYSTEM IMPLEMENTATION

Extensive growth of complexity of software results in the fact that their design, realization, and especially maintenance turn out to be extremely expensive and connected with essential man hours and high requirements to qualification of their developers/maintaining specialists. Statistics states that during the life cycle of software approximately one third of all efforts is spent on their development, and two thirds, on maintenance — the software becomes obsolete, since the requirements to the program of different users and the operation conditions change, which requires its permanent modification. Usually maintenance means update of the software code. For reducing the labor intensiveness of its modification, the ideas of structural programming, module character, data abstraction, object oriented programming were introduced, and requirements to "good" code structuring (which influences maintenance) were formulated. These solutions, of course, simplified software maintenance, nevertheless development and modification of complicated software is still the difficult process. Interactive 3D-environments belong to the class of complicated software.

These factors resulted in the focus of attention of researchers to methods of development and maintenance based on high level languages and mechanisms of automation of design process reducing to a minimum the change of its code [5].

In addition to facts mentioned above, we can also mention rather a new technology of Cloud Computing. Its basic idea is to provide users with services instead of providing them with versions of software to be installed on their computers. The advantages of the cloud computing technology are widely discussed in literature [6], the additional benefit being a significant increase in maintenance capabilities of software during its life-cycle [7], because all the components of software are accessible to developers (or other persons who have the appropriate authority).

Based on the preceding, the following principles of the software system development are proposed:

1. Division of all developers into groups: domain experts, designers, and programmers. Domain experts and designers has own conceptual bases: a virtual world ontology in terms of which they can design and modify a virtual environment model for a whole life cycle [8]. Programmers

joint to development if facilities of the conceptual bases are not enough (they encode additional functions).

2. Support of functioning of a virtual environment by interpretation of its model and generation by it the virtual world. The interpreter of a model of a virtual world provides decreasing laboriousness of the development and maintenance (due to almost entire exclusion of the labour-intensive stage of coding from the life cycle).

3. Implementation of the software system and use of the application (the virtual interactive environment) as cloud services.

In accordance with the suggested principles the architecture of the software system comprises some components (Fig.1). All the components are divided into two classes: software components and information components [8].

Information components are: the virtual environment ontology (consists of the ontology of objects, the ontology of actions, and the ontology of a scenario); a model formed using the ontology (a model contains the complete description of a virtual environment); multi-media data (figures, 3d-models of the virtual world, and so on); agents (additional and independent software modules intended for realization of some specific functions).

Software components are: the structural editor for domain experts controlled by the ontology for editing a model; the graphical editor for designers; the agent editor is the structural editor for creating and modification of agents (agents are independent modules called by the interpreter for processing); the interpreter for processing and analysis of a model and for forming or changing a virtual world.

## III. INTERPRETER

Architecture of the interpreter consists of a server application and a web-client.

The server application works remotely on an outside computer. A user gets access to it accordance with the SaaS model (Software as a Service). The server application controls all resources, the model, and actions of the user.

The web-client run in a browser on the user's computer and provides interface functions for the server application: generation of the virtual world and implementation of interaction between the user and the virtual world.

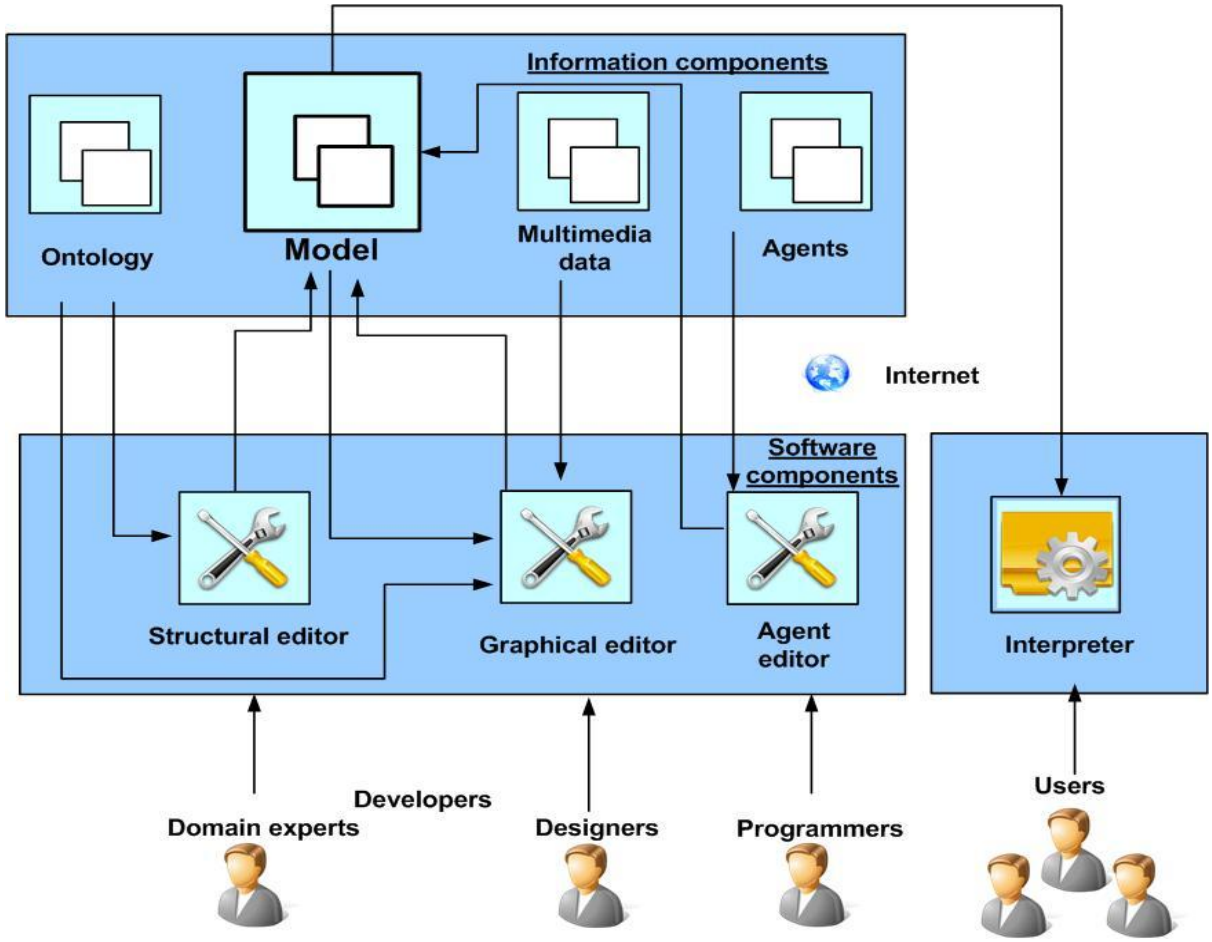


Figure 1. Architecture of the software system

**Server application.** The platform IACPaaS (Intelligenet Application, Control and Platform as a Service) [9] is used for the server application implementation. It provides controlled access and a unified system of administration for development of intelligent services and their components represented by semantic networks and agents, and also supports launching and execution of services. The main architectural components of the IACPaaS platform are: the administrative system, the virtual machine, and the repository. The platform is based on the cloud computing technology and provides the remote access to intelligent systems for users and developers for their development and maintenance.

All the components of the software system are in the repository of the IACPaaS platform (models, the ontology, agents, editors, and the interpreter). The server application is implemented by the multi-agent approach as a set of controlling and processing agents which interacts with other agents by sending and receiving messages. The agents consist of two parts: a declarative one and a procedural one. The declarative part of an agent is its specification (an information resource which describes agent's structure); the procedural part of an agent is a Java code (method) which processes input messages in order to solve the agent's subtask.

**Web-client.** The web-client implemented using the Flash technology. The Flash-file (of the swf format) is saved in the platform repository, and sent and visualized in a browser, embedding to the web-page of the browser as any other media-content (pictures, video-files).

The Flash-technology allows us to generate, visualize, and interact with the virtual world fast enough straightly in a browser using of a computer video-card.

## IV. VIRTUAL ENVIRONMENT MODEL

The virtual environment model comprises three parts: the model of objects, the model of actions, and the scenario model. There are four classes of the objects [10, 11]: "Simple object", "Mutable object", "Compound object", "Table", and two special types of the objects: "Camera" and "Light". Special objects are used for realistic visualization of a virtual world. In general an object description includes a name, a type of the object, and a set of attributes. Attributes are divided in logical attributes and representation attributes. Logical attributes describe characteristics of objects required for task solving in a certain domain; representation attributes are: coordinates, a model of the object, a texture, a scale, and some other attributes required for representation the object in the virtual world.

The model of actions describes actions that a user may execute with objects of a virtual world. Actions can be interactive and instructive; they may either have a valuation and representation of results or change objects.

The model of a scenario is a chain of actions that a user can execute during his or her work with the program. The scenario can include different types of transitions, branches, cycles, and labels.

The process of forming the model consists of two steps. At the first step a domain expert forms the logical part of the model by the structural editor controlled by the virtual environment ontology. He or she defines objects of the virtual world, their logical attributes, actions and the scenario.

At the second step the logical part of the model is loaded to the graphical editor and a designer forms the graphical representation of the virtual world for the objects formed by the domain expert (Fig. 2).

## V. CONCLUSIONS

The architecture and components of the software system for development of virtual interactive environments as a services are considered at the paper. The software system is implemented as a cloud service at the IACPaaS platform. Existing tools for virtual reality development are oriented for programmers. In contrast to them the described software system allowed us to involve domain experts and designers to the development process. They form the declarative model by structural and graphical editors. In some cases programmers are involved in the process of development for programming special functions (agents). As a result the process of development and maintenance of virtual interactive environments is considerably simplified. Using the software system the following cloud services have been implemented: the training simulator for ophthalmology, the virtual chemical laboratory, and the virtual demonstration of a district of the city.

## ACKNOWLEDGMENT

The research was supported by the Russian Foundation for Basic Research, the projects 14-07-00270a, 13-07-00024-a

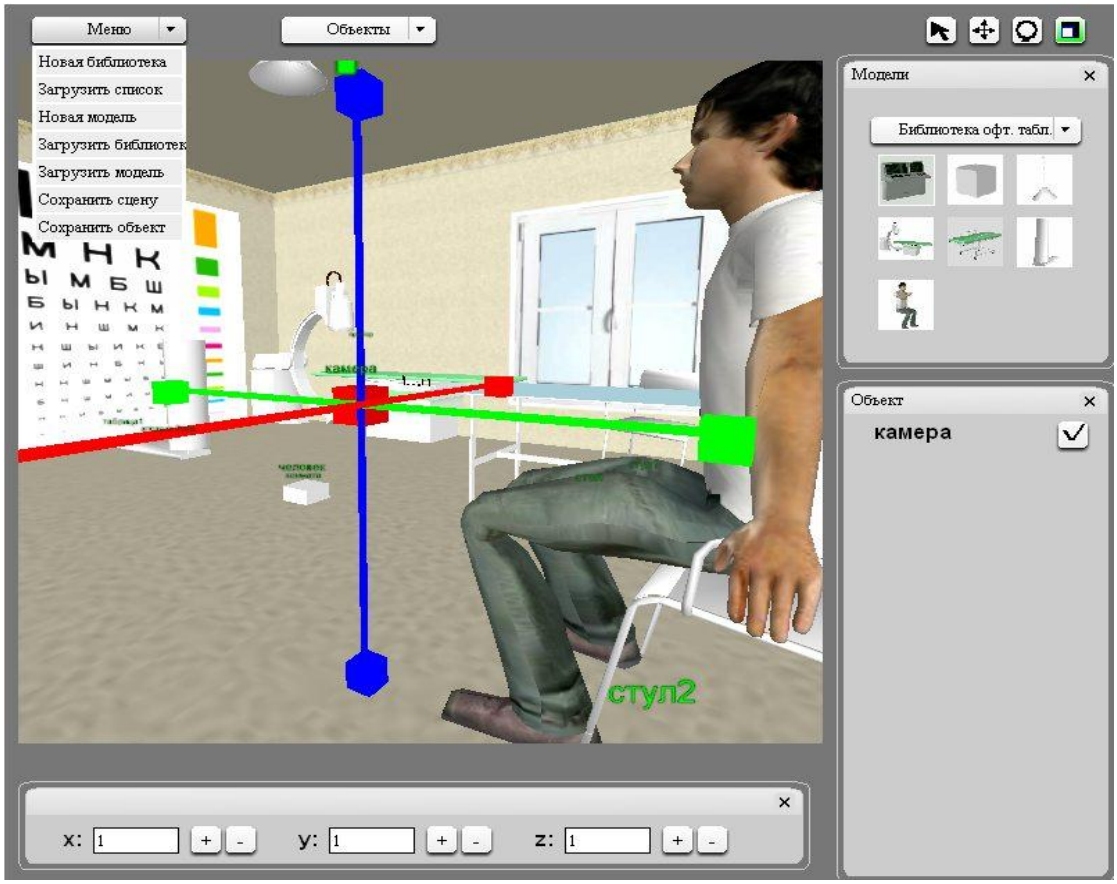


Figure 2. The fragment of the virtual world in the graphical editor

## REFERENCES


[1] Craig A., Sherman W, Will J., Developing Virtual Reality Applications: Foundations of Effective Designpic .– New York: Morgan Kaufmann, 2009. – 448 pp.

[2] VE-Group Virtual reality technologies [Electronic resource] — URL: http://ve-group.ru/

[3] Cruz-Neira C. et al. An Open Source Platform for Virtual Reality Applications Virtual Reality Applications Center Iowa State University [Electronic resource] — URL: http://oldsite.vrjuggler.org/pub/vrjuggler-aiaa2002.pdf

[4] Kuhlen T., Beer T., Gerndt A. The ViSTA Virtual Reality Toolkit. 5th High-End Visualization Workshop, Baton Rouge, Lousianna [Electronic resource] — URL:

[5] http://www.rz.rwthaachen.de/aw/cms/rz/Themen/Virtuelle_Realitaet/research/~piz/vr_software_vista/?lang=de

[6] Chernyak L. E. Adaptability and Adaptivity, Otkryt. Sist., № 9. – 2004 // http://www.osp.ru/os/2004/09/184560.

[7] Gillam L. Cloud Computing: Principles, Systems and Applications // Computer Communications and Networks. - L.: Springer, 2010. – 379 p. - ISBN 9781849962407.

[8] Gribova V.V., Kleshchev A.S., Shalfeeva E.A. Control of Intelligent Systems // Journal of Computer and Systems Sciences International. 2010. Vol. 49. No. 6. Pp. 952–966. - ISSN 1064-2307.

[9] Gribova V.V., Fedorischev L.A. The architecture of Internet software environment for creating teachware with virtual reality // Emerging Intelligent Computing Technology and Applications. – Springer Berlin Heidelberg, 2012. Vol. 304, № 11. – Pp. 394-399.

[10] Valeria Gribova, Aleksander Kleschev, Dmitry Krylov, Philip Moskalenko, Vadim Timchenko, Elena Shalfeyeva, Michael Goldstein. A software platform for the development of intelligent multi-agent internet-services // Proceedings of the Distributed Intelligent Systems and Technologies Workshop (DIST'2013). 1-4 July 2013. St. Petersburg, Russia. – Pp. 29-36. ISBN 978-5-906078-81-0Norvig P., Cohn D. Adaptive software // http://norvig.com/adapaper-pcai.html

[11] Gribova V.V., Petryaeva M.V., Fedorischev L.A. Development of the virtual world for the medical computer simulator // Distance and Virtual Education № 9, pp.56-66 (2011)..