A Cloud Platform for the Development and Use of Intelligent Agent-based Internet-services

Valeria V. Gribova, Aleksander S. Kleschev, Dmitry A. Krylov, Philip M. Moskalenko, Vadim A. Timchenko, Elena A. Shalfeyeva

Abstract--The paper presents a concept of a platform for cloud computing which is intended for development, control and usage of intelligent multi-agent Internet-services. The main purpose of the platform is to decrease the efforts needed for support of such services during their lifecycle.

Index Terms--Agent-oriented programming, cloud computing, internet-services.

I. INTRODUCTION

A large amount of works has been devoted to a problem of creation of intelligent systems (IS) (also called knowledge-based systems). The process of development and maintenance of such systems (which generally consist of a problem-solver, a user interface and a base of knowledge) is difficult and time-consuming. There is still a vast demand on tools for development of IS and on a possibility to reuse their components. An important task is to provide an access to IS and tools of their development. A modern way of providing a convenient access to software systems is based on a concept of “cloud computing”.

Powerful editors (including “cloud” ones) for information content of IS have been developed to date. Their components might be used for creation of new IS [1], [2]. Cloud platforms for software systems of general purpose have been developed and are widely used [3]-[5]. They support separate stages of the development cycle for cloud software systems.

However there are no cloud platforms which fully support the development of all three components of an IS – a knowledge base, a problem-solver and an intelligent interface. Thus there is no support for the whole development cycle of an IS. Most of the existing cloud editors for knowledge bases are limited by a mechanism of “bottom-to-top editing” (from elementary concepts to composite ones) and they do not define the order of knowledge formation, which doesn’t allow domain experts to create and maintain complex structured knowledge bases without the help of mediators (knowledge engineers) in their usual systems of concepts.

Thus, a task of development of a cloud platform which includes tools for support of development of all components of cloud intelligent systems and for their integration is essential.

The paper proposes a cloud-computing platform as a tool to solve these problems. It supports the following technological concepts for development, maintenance and usage of intelligent systems: (1) all information resources (ontologies, knowledge, data) have the same unified declarative representation (a semantic network) [6]; (2) formation and maintenance of knowledge are performed by domain experts and are based on their ontologies; (3) user interface for a domain expert is generated by ontology; (4) method of problem solving is segmented and each part is represented with an agent; (5) high-level API for access to information resources is provided; (6) intelligent system is granted to a user as an internet-service.

II. A PLATFORM FOR INTERNET-SERVICE DEVELOPMENT

A software platform for internet-service development is a software-information internet-complex IACPaaS (Intelligence Application, Control and Platform as a Service) [7]. It provides a controlled access and integrated administrative system for usage and development of intelligent services and their components which are represented with semantic networks, as well as software infrastructure for agent interaction (by message transfer and methods’ initiation and running). IACPaaS is based on a cloud-computing technology and provides a remote access to intelligent systems (for users) and to tools of development and control of such systems (for their developers and controllers). IACPaaS consists of three components: Website, Data warehouse, Virtual machine – see Fig.1.

Website consists of a website engine (CMS “MediaWiki”) and a system service of the IACPaaS platform – “Administrative system”. It is implemented as a set of administrative tools and information resources that are stored in the Data warehouse.

This work was supported by RFBR (research project 12-07-00179-a and research project 13-07-00024-a) and FEB RAS (research project 12-I-P15-03).

V. V. Gribova is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(Academician, e-mail: gribova@iacp.dvo.ru)

A. S. Kleschev is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(DSc, leading researcher, e-mail: kleschev@iacp.dvo.ru)

D. A. Krylov is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(PhD, researcher, e-mail: dmailk@gmail.com)

Ph. M. Moskalenko is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(PhD, researcher, e-mail: philipmm@iacp.dvo.ru)

V. A. Timchenko is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(PhD, researcher, e-mail: vadim@iacp.dvo.ru)

E. A. Shalfeyeva is with the Institute for Automation and Control Processes, Russian Academy of Sciences, Russian Federation.

(PhD, senior researcher, e-mail: shall@iacp.dvo.ru)

ISBN: 978-0-9803267-6-5
Website allows its users: (1) to view the structure of the data warehouse, read news and notifications; (2) to create requests for registration in domains, for acquiring authorities for usage of system and applied internet-services, for modification of the data warehouse; (3) to run their authorities.

The processor of user interface (PUI) consists of a set of interface agents and of an interface interpreter. Interface agents build an abstract user interface in accordance with the ontology of abstract interface. The interface interpreter builds a concrete user interface which is based on the abstract interface and the information received from the problem-solver.

Four conceptual levels can be distinguished in IACPaaS platform – see Fig.3.

The processor of information resources (PIR) is an API for working with inforesources, which are stored in Data warehouse. It is available for developers and supporters of the virtual machine and of software components of services.

The processor of problem-solvers (PPS) provides a launch of the system service “Administrative system” and of users’ authorities, their execution (by activation of agents and transition of messages between agents) and termination.

Fig.1. Components of IACPaaS internet-complex.

Fig.2. Interconnections between processors of virtual machine and components of an intelligent system.

Fig.3. A conceptual architecture of IACPaaS platform.

1. System level (virtual machine level). It is a level of storage and access to inforesources, of launch and execution of services and interaction with user. Functionality of this level is used by services and by maintenance utilities.

2. Library level. It is a level which stores services’ components – agents, message templates and inforesources.

3. Service level. Services as sets of interacting agents are introduced at this level. There are applied services (created by

The data warehouse is a set of storage units which are software and information resources of different types. It is divided into domains which are divided into sections. A section contains storage units: intelligent services, instrumental tools for development and control, their agents, information resources.

Information resources (inforesources) are either problem-oriented or problem-independent. First ones are intelligent services, ontologies and metainontologies of domains, knowledge bases and databases. The second ones are system services which are used for development and maintenance of intelligent services for different domains, problem-independent inforesources (ex.: ontology of user interface). Software components (agents) are also presented as inforesources which have particular content.

The virtual machine is a set of three processors (a processor of information resources, a processor of problem-solvers, and a processor of user interface), and a set of utilities. Each processor is a set of functions for support of correspondent component of intelligent internet-service – see Fig.2.

The virtual machine is a set of storage units which are software and information resources of different types. It is divided into domains which are divided into sections. A section contains storage units: intelligent services, instrumental tools for development and control, their agents, information resources.

Information resources (inforesources) are either problem-oriented or problem-independent. First ones are intelligent services, ontologies and metainontologies of domains, knowledge bases and databases. The second ones are system services which are used for development and maintenance of intelligent services for different domains, problem-independent inforesources (ex.: ontology of user interface). Software components (agents) are also presented as inforesources which have particular content.

The virtual machine is a set of three processors (a processor of information resources, a processor of problem-solvers, and a processor of user interface), and a set of utilities. Each processor is a set of functions for support of correspondent component of intelligent internet-service – see Fig.2.

The virtual machine is a set of storage units which are software and information resources of different types. It is divided into domains which are divided into sections. A section contains storage units: intelligent services, instrumental tools for development and control, their agents, information resources.

Information resources (inforesources) are either problem-oriented or problem-independent. First ones are intelligent services, ontologies and metainontologies of domains, knowledge bases and databases. The second ones are system services which are used for development and maintenance of intelligent services for different domains, problem-independent inforesources (ex.: ontology of user interface). Software components (agents) are also presented as inforesources which have particular content.

The virtual machine is a set of three processors (a processor of information resources, a processor of problem-solvers, and a processor of user interface), and a set of utilities. Each processor is a set of functions for support of correspondent component of intelligent internet-service – see Fig.2.

The virtual machine is a set of storage units which are software and information resources of different types. It is divided into domains which are divided into sections. A section contains storage units: intelligent services, instrumental tools for development and control, their agents, information resources.

Information resources (inforesources) are either problem-oriented or problem-independent. First ones are intelligent services, ontologies and metainontologies of domains, knowledge bases and databases. The second ones are system services which are used for development and maintenance of intelligent services for different domains, problem-independent inforesources (ex.: ontology of user interface). Software components (agents) are also presented as inforesources which have particular content.

The virtual machine is a set of three processors (a processor of information resources, a processor of problem-solvers, and a processor of user interface), and a set of utilities. Each processor is a set of functions for support of correspondent component of intelligent internet-service – see Fig.2.
application developers for solving tasks of users) and there are system services (needed for platform functioning and Data warehouse development). System services are the following:

3.1. Editor of inforesources. It is used for creation and maintenance of invariant (in relation to various technological spaces) complex-structured information of different levels of abstraction by carriers of such information.

The main feature of this approach is to provide a division of labour between carriers of metainformation (knowledge engineers, linguists) who form ontologies with the use of interface which is oriented on metalanguage and carriers of object information (experts, domain specialists) who form knowledge bases and databases with the use of a generated by a particular ontology interface (which is built of customary terms).

3.2. Generator of agents. It is used for generation of templates of code for agents and templates of input and output messages that are used by them basing on their declarative description. It is also used for acquiring the byte-code of reusable message templates.

3.3. Loader of agents. It is used for loading/updating byte-code of agents and templates of input and output messages used by these agents to Data warehouse.

These services (including the Editor of information resources) are meant for providing of instrumental support of stages of life-cycle of agents of IACPaaS platform. At the stage of design the Editor of information resources is used for formation of declarative descriptions of agents and message templates in Data warehouse. At the stage of implementation the Generator of agents is used. It forms an archive of packed files which contain generated classes. At the stage of deployment the Agent loader is used for loading of compiled byte-code of an agent and its message templates to their correspondent storage units, which are formed at the stage of design.

3.4. Agent tester. It is a tool for automation of the process of testing for agents of IACPaaS platform which allows monitoring their state and performing needed control. This service provides a multiple execution of an agent on a given set of tests. Each time it forms and saves a report on the execution.

3.5. Editor of services. It is used for forming/modification of declarative descriptions (specifications) of services in accordance with metainformation of these specifications.

4. Administrative system. It is a tool for providing a controlled access to functionality of IACPaaS platform and management of access rights to usage of applied services, which are developed with the use of the platform.

III. INFORMATION RESOURCES

Knowledge in intelligent systems has its own value. It can be created and used independently from the tools of its processing by both – people and computer programs. It must also be kept up to date over a long period of time. A representation of any type of information (metainformation, ontology, knowledge, data) in the Data warehouse is called an information resource.

All information resources have the same unified declarative representation – a hierarchical semantic network. The main feature of information resources is that they are represented as a pair: object information – a link to metainformation. Metainformation of an inforesource is a language in terms of which object information is formed. Such language is understood by domain experts. This allows them to create and maintain information resources without mediators (knowledge engineers). The metainformation is formed in accordance with the model of structural reusable representation of information resources.

IV. API OF INFORMATION RESOURCES

The main task of this component (which is a part of PIR) is to provide a set of programming interfaces for access to information resources for a developer of a service. Thus developers can perform various permitted operations on inforesources without thinking of the details of internal information representation and operation’s implementation.

This API provides a controlled access to information resources, controls the correspondence between metainformation and object information and maintains integrity and completeness of the object information. The main groups of API functions are: functions for working with concepts and relations of the network (acquisition and edition of the values of attributes, generation), functions for checking various conditions, functions for navigation through network etc.

V. AGENT-BASED INTERNET-SERVICES

An intelligent agent-based Internet-service is represented with its declarative description which generally includes: (1) input, output and own information resources (they might be ontologies, knowledge bases, databases, etc.), (2) problem-solver which is represented by a description of set of agents and maybe a control graph, (3) user interface (see Fig.4.).

![Fig.4. Architectural context scheme of a service.](image-url)
VI. INPUT, OUTPUT, OWN INFORMATION RESOURCES

Input and output information resources of the service are its input and output formal parameters which define classes of processed information resources. The values of these parameters are information resources for which the formal parameter resources are ontologies. The information resources that are the values of the input parameters cannot be modified. The information resources that are the values of the output parameters can be changed while being processed by agents of the service. Own information resources are used for storing own data which may include service settings that control its work. It must be noted that any of these types of information resources can be absent in a service.

VII. PROBLEM-SOLVER

A problem-solver of an intelligent internet-service is represented with a set of agents and can include a control graph. Among this set a single agent must be a root one. The platform sends an initializing message to this agent thus starting the work of the whole service. In case a service has a user interface – a special single interface-controller agent must be implemented. It works as a receiver of messages from system agent called “View”, which sends them in case a user generates events in the interface.

VIII. AGENTS

An agent is a reusable software component which interacts with other agents by sending and receiving messages [8], [9]. Reusability means that an agent can be a component of different services. Generally an agent consists 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. It is used for automatic generation of template of agent’s source code by Generator of agents. The procedural part of an agent (its code) is a byte-code, obtained by compilation of agent’s source code and stored in its declarative description with the use of system service Loader of agents.

An agent includes blocks of productions (methods) which perform processing of received messages in order to solve agent’s task. For each block of productions a message template is determined. The result of message’s processing might be a formation and sending of one or several messages to one or several agents and formation or modification of some information resource (own or output one).

A message is a temporary information resource. Its lifecycle starts when it is created by some agent by some template. Then it is sent to another agent which receives and processes it. After that the message ceases to exist. Message templates determine the language of agent interaction, so an agent should “know” the template of the message in order to be able to send it to another agent. An agent can send a message: by return address (to the agent from which it has received a message which initiated a block of productions), to an agent which is specified in that initiating message, to an agent which is specified in control graph, to an agent which is specified in the code of block of productions.

IX. CONTROL GRAPH

A control graph is used when problem-independent agents are included into the problem-solver of the service. Such agents should not send messages directly to others. In this case a control graph is used by the processor of problem-solvers for determining an agent to which a message should be sent. The control graph consists of nodes (which correspond to blocks of production of agents) and of arcs (which correspond to message sending between agents). A node of a control graph has a unique name and stores a link to the agent of problem-solver which includes the block of productions represented by that node. An arc has a unique label and contains a link to a template of the transferred message and links to its sender and receiver.

X. USER INTERFACE

In order to simplify the process of development of user interface a system agent “View” was included into the IACPaaS platform. It interacts with the Web-server, processing agents (which interact with interface agents) and with the user interface interpreter in the following way: (1) a request to Web-server is formed through a client-side software (browser); (2) the Web-server processes the received information and transfers it to the “View” agent; (3) the “View” agent creates a message, and forms its content with the received data and sends it to a processing agent of problem-solver (to its interface controller); (4) the interface controller receives the message and performs required actions (with possible interactions with other agents of problem-solver).

When the results of message processing are to be shown to user: (1) the processing agent forms a description of an abstract interface (with the help of interface agents and in accordance with the ontology of abstract interface1) and sends it in a message to “View” agent; (2) the “View” agent receives this message, forms a description of a fragment of concrete interface on a basis of abstract interface and a model of concrete interface (using the interface interpreter) and sends it to Web-server; (3) Web-server forms a reply to the browser which displays a fragment of the user interface.

XI. CONCLUSION

The paper presents a concept of the platform for development and usage of intelligent internet-services. The main goal of platform creation is to decrease with its usage the efforts needed for development and maintenance of internet-services.

In order to achieve this goal all information resources of intelligent service are presented in a single unified way – in the form of a hierarchical semantic network. At the same time a

1 An ontology of an abstract interface contains a description of standard interface elements: label, control button, icon-button, input text field, checkbox, radio-button, drop-down list, link, etc. Some interface elements can produce events in response to user’s actions.
unified API is provided for access and modification of all components of intelligent service. A representation of problem-solver as a set of agents and a unified format of knowledge base and database representation allows one to reuse these components for creation of different intelligent internet-services. Knowledge bases are formed on a basis of their ontologies. It allows domain experts to create and modify such bases by themselves (without mediators and additional tutoring).

The usage of cloud platform IACPaaS for development of internet-services meets modern requirements for design and implementation of software systems and provides their viability (adaptability and controllability) by the fact that all components of the service are available to developers for maintaining in actual state. The development of an agent is reduced to specification of the components of its declarative part as well as to writing the procedural code of consequents for set of blocks of productions. The development of a service is reduced to specification of its declarative component and its maintenance is reduced to modification of this component.

REFERENCES