MODELING OF AN INTELLECTUAL PROBLEM SOLVER BY TRANSFORMATION OF SEMANTIC MODELS

Elena Shalfeeva

***Abstract**: The method of construction of a problem solver of intelligent software by transformation of its models is presented. The representations of all these models are convenient both for the person and for program processing. Based on this method supporting tools for intelligent software engineering will be constructed.*

***Keywords**: intelligent software, problem solver, maintenance, hierarchical semantic network.*

***ACM Classification Keywords**: D.2.11 Software Architectures (Domain-specific architectures)*

Introduction

An automation of intelligent professional activity is widespread sphere of application of efforts of IT specialists. Modern software systems optimize circulation of documents, keep account expenses, form reports and personnel work records. However such program complexes don't include any tools of support of experts' decision-making [Белоцерковцева, 2010].

Successes in automation of support of decision-making of intellectual tasks are related to knowledge-based systems. But in spite of considerable successes in area of the theory and technology of their creation, they didn't take noticeable practical use. As a rule, such systems are created within research, innovative program projects and don't develop further in view of viability problems [Клещев, 2011].

Nowdays the creation of the integrated complex of problem solvers, knowledge bases and tools of their maintenance is expected from automation of an intelligent professional activity [Клещев, 2011; Румянцева, 2007]. Some subsystem which solves some intellectual problem or gives support in decision-making, using the knowledge base, is called here as a problem solver. The integrated complex has to be scalable and controlled [Клещев, 2010]. It demands the integration of the ideas from artificial intelligence with the approaches of modern programming [Шалфеева, 2013; Грибова, 2012]. The new approach to development of the controlled long-living intelligent software with conceptual knowledge base is presented in the [Клещев, 2010]. It is based on the hypothesis of separation of declarative and procedural components in specialized solvers.

The purpose of this research - to develop a method of transformation of domain and functional models of a specialized intellectual problem solver to its declarative design representations. Such method is necessary for creation of tools for development of maintainable intelligent software (ISW).

The Creation of functionality models of a solver

The result of system analysis (for automation of intelligent professional activity) are the set of models of subject domain, system and its subsystems [Клещев, 2012], among them – the domain ontology, the mathematical statement of each intellectual problem, algorithm of execution of each such problem. Domain ontology consists of two parts: “reality ontology” (the structure of the input and output data) and “knowledge ontology” (the structure of knowledge used by experts).

A developer of mathematical statement of problem, as a rule, defines a method of the problem solution. The example of method of diagnostics in medicine – "search of all possible values of the diseases; for each disease the following is carried out: forming of all possible variants of each cause-and-effect relation development and searching among all of them the variants corresponded to values of patient signs” [Москаленко, 2006]. The analyst or knowledge engineer builds the algorithm of the execution in according to the method of problem solution and the knowledge ontology. The most traditional representation for algorithm is the flowchart. This algorithm can be presented and as step-by-step transition from a compose task to elementary subtasks.

An example. The fragment of the algorithm for diagnostics [Москаленко, 2006] can be presented so:

“...to check the hypothesis “the patient is healthy“;

(to organize the Cycle on observed signs;

((to check the hypothesis “all observed values of the sign can take place at any healthy patient;

to record the value of a sign, if it disproves the hypothesis))

to sum up the hypothesis analysis);...”.

Before designing of each subsystem the detailed analysis of its functions is carried out. The functional model of any problem solver can depend on the user requirements also.

Universal approach to software functionality analysis is partitioning [Pressman, 2001]. The subtask of some level (consecutive or having some options of possible actions) breaks into some less complex subtasks. For some of them it can be necessary "cyclic address". Thus, the hierarchical model of subtasks is the reflex of algorithm contents with addition of input and output subfunctions, and probably subfunctions providing some user requirements.

The variant of representation of this model is following. It is advisability to keep hierarchical model of solver’s subtasks (HMsT) in the form of a hierarchical semantic network (HSN).

During partitioning it is important to distinguish, whether element of hierarchy of some level consists in "simple" submission or in "cyclic" submission, and whether it will be "optional" [Mori, 2002].

Meta-infomation for HMsT (i.e. the structure of HSN containing HMsT) can be described as follows (alternative elements of a network, nonempty and any sets of elements are marked here with qualifiers “alt”, “set” and “setmm”):

task {

name: string;

structuredness: ~alt (the consecutive; with - options)

~set subtask

{ subordination: ~alt (in "simple" submission;

in "cyclic" submission;

"chosen on a condition")

название: строка;

-> structuredness;

[~setmm {–> subtask }] }}.

Traditionally the software requirements analysis presents an associations of functions with received and modified elements of information. For the accounting of all processed data of a subsystem every used (or modified) data can be classified to one of three types: stored, received interactively, formed during decision and not-stored.

The expanded model of subtasks (EMsT) enriches HMsT with associations of data. The variant of representation of this model (meta-infomation for EMsT) is following.

task {

name: string;

structuredness: ~alt (consecutive; with options)

~set subtask

{ subordination: ~alt (in "simple" submission;

in "cyclic" submission;

"be chosen on a condition")

name: string;

-> structuredness;

~set used data { data name: string;

availability: ~alt { stored, interactive, in process};

~set modified data { -> data name; -> availability };

~setmm {–> subtask }}}.

An example. The HSN fragment for EMsT of a solver of medical diagnostics is presented in the next figure.

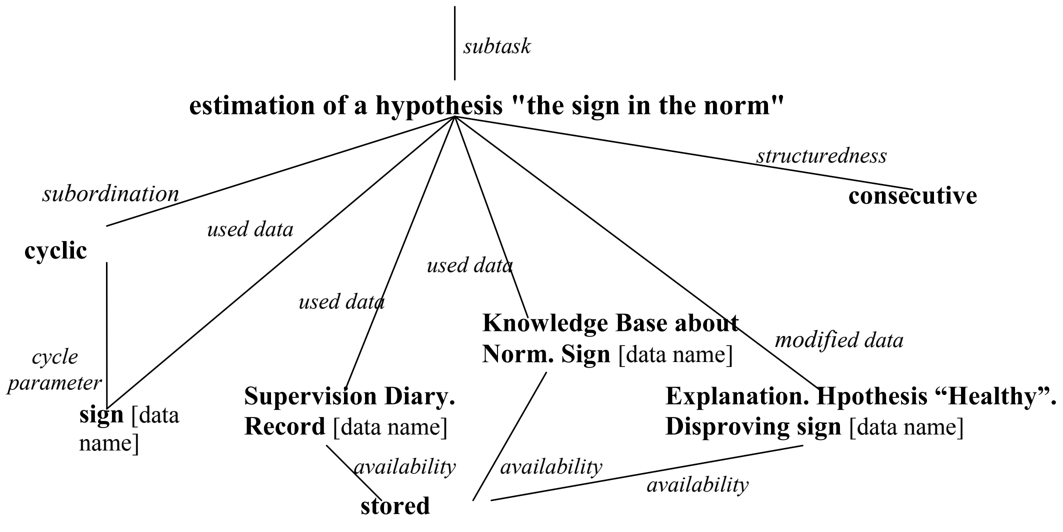


Figure 1. The HSN fragment for EMsT for a subtask of estimation of a hypothesis "the sign in the norm".

The Creation of architectural representations of a solver

Software system Designing includes the construction of architectural representations of each subsystem.

In the course of architectural design of rather independent subsystem (such as the solver) designers focus attention, first, on modular decomposition and functionality distribution between these modules (units), and secondly, on control modeling – definition of interaction between units [Pressman, 2001]. Quality of the solver realization and possibility of its maintenance depends on these architectural models [Pressman, 2001, Басс

2006]. Procedures, functions, objects\classes, agents or others can become the units of a specialized solver, depending on approach to its realization.

The construction of a specialized solver by means of call-return architecture can be carried out by known design methods [Pressman, 2001]. In the other approach to architectural designing of the specialized solver processed knowledge and data in the form of ISS, there are two types of units. The first type – the active, rather independent units communicated each other by means of messages (let's call them agents [Грибова, 2012]), the second type – the operations over classes of semantic networks. For such architecture at first the model of necessary is defined agents (according to models of subtasks), then the interactions between units are modelled, further necessary operations are designed.

Model of all needed units. The functionality (presented by HMsT) has to be distributed by the corresponding quantity of units. It is specified how agents involve each other for cooperation. The agent can be made of two or more structural units (blocks) corresponding to strongly coupled subfunctions.

The variant of representation of this model is following (for above-designated "agents-approach").

network of agents {

root agent :string;

~ set cooperation {

agent-collaborator name :string;{

agent type ~alt (coordinating, processing, grouping)

~ setmm ->cooperation } } }

The method of models transformation: root agent is confronted with the main task; agents with two blocks are confronted with "composite" tasks, processing agents - with “leaf” subtasks. For optimization of units number some “leaf” subtasks can be realized as complementary blocks of certain agents.

<u>An example</u>. The HSN fragment for “network of agents” of medical diagnostics solver is presented in the next figure 2.

Here unit for Hypothesis “Healthy" for own performance has to check "normality" of all relevant signs, therefore it (for each of signs) addresses to unit “estimation of a hypothesis "a sign in the norm". After all patient’s signs will be estimated, the general conclusion is formulated, whether he is healthy.

Model of control flow between units. The model of agents communication by messages (called by control graph) is intended for its use at a runtime: the processor gets to know about the recipient of the next message from this model. By changing of elements of this model, it is possible to influence a solver’s configuration.

The model of control flow between units is being constructed on base of model of subtasks and a network of agents. The method of control graph construction can be follows. For every communication specification one of reusable message templates is selected (or the new template is specificated and designed). The agents, confronted to subordinated subtasks, receive the message by “task” template. The agent for a subtask in "simple" subordination, as a rule, forms the response message on the “feedback” template, and in "cyclic" subordination – "for cycle communication" template.

The most convenient representation of control flow between units for person’s perception is in the form of graph. However HSN- representation is needed to support uniform storage of all models of a solver. For example, the representation of control graph nodes may be following.

~ set agent’s block node{

block name :string;

name of the agent :string;

launching message template : $ template}.

An example. Fig. 3 shows some fragment of the control graph defining a scheme for transmitting messages between agents for medical diagnostics problem solver. The agents with several blocks are depicted with multiple entry points.

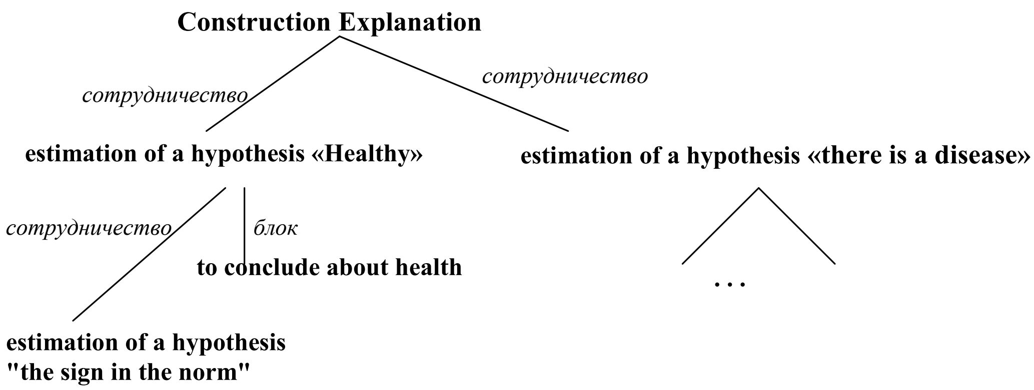


Figure 2. The fragment of the model of all needed units for medical diagnostics solver

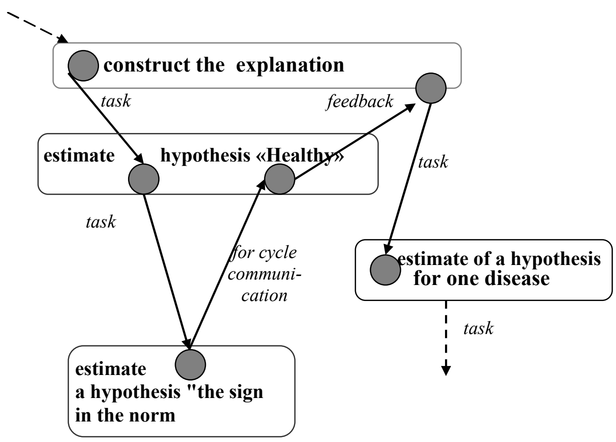


Figure3. The fragment of the control graph for medical diagnostics problem solver

Further designing is aimed to addition in architectural model of information communications of units, associations with information resources (IR). It is aimed to designing of separate units (adapted for a reuse). For ensuring quality of each solver of ISW it is required to construct such models as the design of information communications of agents, the specifications and interfaces design of each agent and designs of each IR with sets of operations of access to these stored IR.

Conclusion

For increase of the maintainability of systems, automated the intelligent activity, the new approach to the development of each of its subsystems (intellectual problem solver) has proposed.

This approach is relevant for systems in which knowledge base, the first, are controlled by the experts, secondly, are stored in the conceptual resources (semantic networks), and, the third, are available by software.

This approach consists of the representation of results of problem solver development in the form of the declarative models clear to the developer. Architectural models of solvers under construction are realized within the declarative agent programming [Грибова, 2012]. Decision-making agents, which process data or use stored knowledge, have means to access to the semantic networks storing data and knowledge.

The retention of declarative models in the form of semantic networks provides convenient program access to them. Existence of such access to all solver models and existence of a method of transformation of one (semantic) models for construction of the following models are base for creation of those intelligent systems development tools components, which support designing of solvers.

Acknowledgements

This work is performed with financial support of grants of the RFBR № 12-07-00179-а and FEB RUS № 12-I-ОНИТ-04.

Bibliography

[Mori, 2002] Mori G., Paternò F., Santoro C.: CTTE: support for developing and analyzing task models for interactive system design. IEEE Trans. Softw. Eng. 2002, № 28(8). С. 797–813.

[Pressman, 2001] Pressman R.S. Software Engineering: Practitioner's Approach. Fifth edition. McGraw-Hill Inc., 2001. 860 p.

[Басс 2006] Басс Л., Клементс П., Кацман Р. Архитектура программного обеспечения на практике. 2-е изд. – Л.:Питер, 2006. 576 с.

[Белоцерковцева, 2010] Белоцерковцева Л.Д. Опыт автоматизации деятельности медперсонала перинатального центра города Сургута // Федеральный справочник "Здравоохранение России". Т. 9. 2010. С. 469-474.

[Грибова, 2012] Грибова В.В., Клещев А.С., Крылов Д.А., Москаленко Ф.М., Тимченко В.А., Шалфеева Е.А. Агентный подход к разработке интеллектуальных интернет-сервисов // Труды Конгресса по интеллектуальным системам и информационным технологиям «IS&IT'12». М.: Физматлит, 2012. Т.1. с. 218-223.

[Клещев, 2010] Клещев А.С., Грибова В.В. Управление интеллектуальными системами. Известия РАН. Теории и системы управления. 2010. № 6. С. 122-137.

[Клещев, 2011] Клещев А.С., Грибова В.В. Проблемы создания жизнеспособных интеллектуальных систем и методы их решения // International Journal "Information Technologies & Knowledge" Vol.5, Number 3, 2011. С. 250-258.

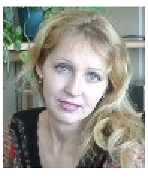
[Клещев, 2012] Клещев А.С., Шалфеева Е.А. Системный анализ при автоматизации интеллектуальной профессиональной деятельности // Труды XIII Национальной конференции по искусственному интеллекту с международным участием «КИИ-2012», Белгород: Изд-во БГТУ, 2012. т.2. С.128–135.

[Москаленко, 2006] Москаленко Ф.М. Алгоритм диагностики, основанный на реальной онтологии медицины, для многопроцессорной ЭВМ / Доклад Paco2006, 2006. Издатель: Институт проблем управления им. В.А. Трапезникова РАН.

[Румянцева, 2007] Румянцева Е.Л., Слюсарь В.В. Информационные технологии. Общая характеристика систем поддержки принятия решений // М.: Форум, Инфра-М, 2007.

[Шалфеева, 2013] Шалфеева Е.А. Семантические модели представления решателя Интеллектуальной системы // Материалы III Междунар. научн.-техн. конф. "Открытые семантические технологии проектирования интеллектуальных систем" (OSTIS-2013) / отв. ред.: В. В. Голенков. Минск: БГУИР, 2013. С. 257-264.

Authors' Information



***Elena Shalfeeva** – senior researcher of Intelligent system laboratory, IACP, Vladivostok, Russia; e-mail: shalf@iacp.dvo.ru*

Major Fields of Scientific Research: artificial Intelligence and decision making, program models and systems