Modeling of a Procedural Knowledge by a Dialogue Knowledge Base

Igor Chimir [1] and Waheeb Abu-Dawwas [2]
[1] Cherkassy Academy of Management, Ukraine
[2] Computer Information Systems Department, Al-Zaytoonah University, Jordan

Abstract: *This paper depicts theoretical results obtained in the line of projects related to constructing dialogue applications based on a formal cognitive model of a human-machine dialogue. One of the aims of the paper is to propose an appropriate model of question-answering dialogue, which can be used in designing relevant computer software. The theory proposes formal descriptions of declarative and procedural knowledge of dialogue's agents and introduces the idea of a dialogue knowledge base, which is capable of storing the procedural and the declarative knowledge of dialogue's agents. Emphasis on declarative-procedural typology of knowledge, allows us to consider a dialogue process as a goal-oriented behavior; and, hence, as a general method of solving some classes of problems.*

Keywords: *Human-machine dialogue, question-answering dialogue, logic of questions and answers, dialogue knowledge base, problem solving process.*

Received April 21, 2003; accepted July 27, 2003

1. Introduction

Declarative vs. procedural dichotomy of knowledge has an influence on practically all key aspects of cognitive science, e.g: *teaching* (teaching of skills usually starts from the knowledge of declarative representation of a target domain and continues by teaching the ability to manipulate an acquired declarative knowledge in order to achieve a goal); *automatic and conscious processes* (acquiring of certain skills entails transition of the procedural knowledge into the rank of automatic, when initial declarative knowledge declines, and the concept "automatic process" becomes a synonym of the concept "procedural knowledge"); *schema* (a schema can be considered as a "keeper" of a chunk of declarative knowledge in the form of properties and relations between chunks); *memory* (a set of linked schemata stores in memory, which represents both types of knowledge, and where links between schemata simulate procedural knowledge); *sensory system* (a sensory system plays a significant role in forming the initial collection of elements of declarative knowledge, whereas conscious processes are mainly responsible for forming a procedural knowledge in the form of links between schemata).

Representation of a dialogue process from the point of view of declarative-procedural typology can be helpful in building a theory of solving ill-formalized problems by means of dialogue methods. Two main ideas are at the heart of the theory: logical structure of question-answering pairs, and conception of a dialogue knowledge base, which can store knowledge of a goal-oriented behavior of both dialogue agents.

2. Declarative-Procedural Typology of Knowledge in the Dialogue Process

Separation of knowledge as *declarative* and *procedural* is a generally adopted classification and is a basis for models of memory and problem solving processes [8, 18]. Declarative-procedural distinction of knowledge representation is also in the foundation of architecture and behavior of unified cognitive models ACT [1] and SOAR [12].

Declarative knowledge is usually associated with facts or factual knowledge, which can be described verbally, e.g. in the form of propositions such as "a bird is an animal which can fly." To represent a chunk of declarative knowledge in addition to symbolic description, we might also use images, especially in those cases when an image is hard or impossible to be reduced to symbols. There are some ways of modeling of chunks of declarative knowledge: *schemata* [9], *frames* [10], etc. In order to model a system of declarative knowledge we often use *semantic network*, which is a set of declarative knowledge chunks along with a set of relationships between them.

We associate procedural knowledge with abilities and skills, e.g. the ability to ride a bicycle or the ability to type using a blind keyboard. As a rule, to operate procedural knowledge people do not need conscious efforts and do not use an attentional system. The fundamental way of formal modeling of a fragment of procedural knowledge is *production rules* [1, 12].

Declarative-procedural typology of knowledge yields two types of long-term memory organization: a system with two long-term memory types (declarative and procedural), and a system with one universal long-term memory.

The most famous cognitive system based on two long-term memories is a family of models called ACT, and offered by John Anderson [1]. Anderson presupposes that the human's mental system indeed includes two types of long-term memory for keeping declarative and procedural knowledge separate, and implements this supposition into his ACT models.

SOAR cognitive system, proposed by Allen Newell [12], has a lot of features in common with the ACT family, but in contrast to the latter presupposes that a long-term memory should keep both types of knowledge. While solving a problem, SOAR retrieves needed declarative knowledge from its long-term memory and temporarily locates them in the working (short-term) memory. Hence, SOAR's short-term memory accumulates only the declarative knowledge which is relevant to the current problem and will be used to find a solution of the problem.

Two questions arise when we are trying to apply declarative-procedural typology of knowledge to the dialogue process modeling: (1) what kind of semantics are behind the declarative and procedural knowledge concepts in the context of a dialogue? (2) is it a rational idea to keep declarative and procedural knowledge separately within a dialogue system?.

Among a number of problem-oriented and problem-independent theories of dialogue, we are emphasizing a theory of question-answering dialogue [4, 5, 19]. The name of the theory reflects the fact that it based on certain fundamental assertions of the logic of questions and answers [3]. During a question-answering dialogue the messages (in symbolic or non-symbolic form), which dialogue agents send to each other have the status and logical structure of questions and answers. We focus our attention on question-answering dialogue because, as will be shown later in the paper, this type of a dialogue can serve as a problem-solving procedure for some types of problems.

First, we must explain that we consider a dialogue to be a discrete or step-by-step process. We consider a step to be a "behaviorist molecule" of a dialogue and assume that all dialogue scenarios can be created from a finite number of steps. During each step an elementary cycle of agents' knowledge interchange is completed.

The knowledge interchange within a step presupposes an agent's asymmetry. This means that one of the agents initiates the knowledge interchange, and that the second agent responds. Let the agent-initiator of knowledge interchange be called an active agent or *A-agent*, and the opposite agent be called a reactive agent or *R-agent*.

The R-agent is logically dependent on the A-agent. The R-agent is not free in choosing the answer but must return to the A-agent a relevant chunk of knowledge. This is because, in the opposite case, the logic of dialogue is disturbed, and dialogue process is transformed into two independent monologues.

Within a dialogue step the active and reactive agents transmit to each other chunks of declarative knowledge. In a question-answering dialogue, a chunk of declarative knowledge, which A-agent transmits to the R-agent, has the logical structure of a question. A chunk of knowledge that R-agent returns to the A-agent has the logical structure of an answer. We use the term "logical structure" to allow for cases where questions and/or answers have non-symbolic representations.

Let knowledge chunks that have a logical structure of questions and answers be called Q and Achunks, respectively. Results obtained in the logic of questions and answers [3] allow us to state that Q-chunk carries two types of information: a fragment of declarative knowledge from which all answers for the given question can be formed, called the *subject of the question*; and a specification of the desired answer called the *prerequisite of the question*.

The subject is the "raw material" for the answer. The R-agent, while generating the answer, does not use all the accessible declarative knowledge, but only this fragment. The prerequisite determines what part of the Subject should be in the answer. Consequently, general logical structure of Q and A chunks is as follows:

$$def{:}\ Q\text{-}chunk = Pre, Subj \tag{1}$$

$$def{:}\ A\text{-}chunk \in Subj \tag{2}$$

Where Subj and Pre are the Subject and the prerequisite of the question, respectively. We will consider that the Subject is a set of semantically relative elements, and that the prerequisite is an encoded specification of the answer:

$$Q\text{-}chunk = Pre, \{Subj_a\} \quad a = 1 \,..\, m \tag{3}$$

Where: $\{Subj_a\}$ $a = 1 \,..\, m$ set of the Subject elements.

The A-chunk subsequently is a subset of the Subject elements:

$$A\text{-}chunk = \{Subj_a\} \quad a = 1 \,..\, n; \quad n < m \tag{4}$$

Let us consider two simple verbal examples illustrating the concepts of Subject and prerequisite.

1. What prime numbers are between ten and twenty?
2. Give an example of a prime number between ten and twenty?

Both questions have the same subject: {11, 13, 17, 19}, but different prerequisites. The prerequisite in the first question defines a single answer: "Between ten and twenty there are the following prime numbers: 11, 13, 17, 19". The prerequisite in the second question defines several answers: "An example of a prime number between ten and twenty is: 11"; "An example of a prime number between ten and twenty is: 13" and so forth.

Every Q-chunk therefore can yield a *set of possible answers*:

$$A\text{-}set = \{A\text{-}chunk_{\beta}\}\ \ \beta = 1\,.\,.\,k \qquad (5)$$

The A-agent, in accordance with the goal of the dialogue, plans to receive and recognize a more restricted set than the A-set set of answers. Let this set be called the *recognizable set of answers* or RA-set. RA-set unites answers that A-agent needs at the current step. All other answers can be classified as *non-recognizable* or NA answers.

The subject and prerequisite structure of the Q-chunk given above allows us to evaluate possibilities of verbal and nonverbal representation of information within a question. Clearly, at least subject's elements can be represented nonverbally. As for the prerequisite, an explanation might have a verbal representation, either textual or sound.

Formulas (3) and (4) represent the structure of declarative knowledge within a question-answering pair in the form of a set of elements. Such representation is enough on the level of general definitions, but from a practical point of view it seems to be simplified and needs some elaboration. Analysis of question-answering pairs from real question-answering dialogues demonstrates that, as a rule, two entities appear as the subject of a question: (1) a single object, which has a status of a *thing* or *property*; and (2) associated – with this object – a *list of properties or things*, respectively.

Thus, we can state that along with the question A-agent transmits to R-agent a subject of the question, which has one of the following structures:

def: Subj = <object-thing>
{expanded list of properties} (6)

def: Subj = <object-property>
{expanded list of things} (7)

The R-agent constructs the answer by extracting a sub-list from the expanded subject's list. The answer therefore might have one of the following structures.

def: A-chunk = <object-thing> HAS PROPERTIES
{list of properties} (8)

def: A-chunk = <object-property> ATTRIBUTED TO
{list of things} (9)

Example 1

Question: Is glass a liquid when the temperature is 70 F?

Answer: Glass is not a liquid when the temperature is 70 F.

Subject in the form (6): <glass under 70 F>, {<to be a liquid, not to be a liquid>}

Answer in the form (8): <glass under 70 F> HAS PROPERTY <not to be a liquid>

Example 2

Question: What prime numbers are between 10 and 20?

Answer: Between 10 and 20 there are the following prime numbers: 11,13,17,19.

Subject in the form (7): <be a prime number> {11,12,13,14,15,16,17,18,19,20}

Answer in the form (9): <be a prime number> ATTRIBUTED TO {11,13,17,19}

We can rewrite formulas (6) – (9) in strict notation using conceptual basis of first order logic interpreting a property as a one-place predicate. Hence, we can treat subject as the following expression

$$Subj = x, \{P_a(x)\},\ a = 1,\,.\,.\,,m \qquad (10)$$

Where $P_a(x)$ is a one-place predicate, X HAS PROPERTY P_a

Expression (10) is the analogue of expression (6), whereas the analogue of expression (7) is:

$$Subj = P(x), \{x_a\},\ a = 1,\,.\,.\,,m \qquad (11)$$

Where x_α is the value of variable x.

From the R-agent's point of view the subject contains false and true propositions yielded by P(x) predicate. It is worth noticing that truthfulness or falsity of the subject's elements, in the case of question-answering dialogue, are not absolute categories but rather have relative meaning in relation to R-agent's vision of the world. For instance, smoking could be a bad habit for one agent and a pleasure for another. This is a reason why one question yields more than one true answer.

Hence, we can interpret an answer as a question, from an expanded list in which R-agent has eliminated all false elements (of course, in accordance with R-agent's vision of the world.) Therefore, we can use expressions like (10) and (11) to model possible structures of the answer:

$$Ans = x, \{P_a(x)\},\ a = 1,\,.\,.\,,n \qquad (12)$$

$$Ans = P(x), \{x_a\},\ a = 1,\,.\,.\,,n \qquad (13)$$

$$n < m$$

The prerequisite sets the completeness (number of elements n in (12) and (13)) of selection from the question subject's expanded list.). Analysis of examples suggests seven classes of prerequisites as depicted in Table 1.

Table 1. Classes of perquisites.

Classes of prerequisites	**Completeness of answer**	**Natural language formulations**
Pre_1	One element	
Pre_2	Some elements. Exact number of elements is unknown	
Pre_3	Uncertain number of elements from upper to lower bound	"...less then <up.b.> but greater then <lw.b.>..."
Pre_4	Uncertain number of elements from lower bound and up to the whole list	1. "...not less then <lw.b.>..." 2. "...at least <lw.b.>..."
Pre_5	Uncertain number of elements from one element and up to the upper bound	"...at most <up.b.>..."
Pre_6	Uncertain number of elements from one element and up to the whole list	"...at least one ..."
Pre_7	The whole list	

Discussion given above allows us to structure fragments of declarative knowledge, which are relevant to question-answering pairs in the question-answering dialogue by means of two collections of alternative expressions:

$$\left.\begin{array}{l} Q\text{-}chunk = Pre_g,\ x,\ \{P_a(x)\};\ a=1,\ldots,m;\ \ g=1,\ldots,l; \\ A\text{-}set = \{A\text{-}chunk_b\};\ \ b=1,\ldots,k; \\ A\text{-}chunk_b = x,\ \{P_a(x)\};\ a=1,\ldots,n;\ \ n < m \end{array}\right\} \quad (14)$$

$$\left.\begin{array}{l} Q\text{-}chunk = Pre_g,\ P(x),\ \{x_a\};\ a=1,\ldots,m;\ \ g=1,\ldots,l; \\ A\text{-}set = \{A\text{-}chunk_b\};\ \ b=1,\ldots,k; \\ A\text{-}chunk_b = P(x),\ \{x_a\};\ \ a=1,\ldots,n;\ \ n < m \end{array}\right\} \quad (15)$$

Where:

Q-chunk: a question;
A-chunk: an answer;
A-set: a set of possible answers;
Pre_g: g class of prerequisite;
P(x): a one-place predicate X HAS PROPERTY P(x)";
k: number of possible answers;
l: number of classes for prerequisite;
m: number of elements for the expanded list of a question's subject;
n: number of elements for the list of an answer.

Before we start our discussion regarding procedural knowledge in the context of the question-answering dialogue, we have to consider a concept of a *dialogue goal*. Such an approach is probably correct in all cases when we are speculating about procedural knowledge. Procedural knowledge is skill, which is always directed toward achieving a certain goal. We can find confirmation of this claim – for instance – in a number of arguments regarding practical value of the ACT model in which the authors devoted much attention to the concept of goal [2].

We are considering *procedural knowledge of agents* on two levels: (1) the level of a *dialogue scenario;* and (2) the level of a *dialogue step*. Procedural knowledge of an active agent on a scenario level is a strategy of achieving the goal by the active agent. An active agent must know how to direct the dialogue to achieve the goal, which means knowledge of what particular question must be returned to the reactive agent as a response to his current answer. The goal of the active agent on this level is an achievement of an expected (target) answer. A reactive agent, as well as an active one, has its own goal which is obviously not the same as that of the goal of the active agent. Therefore, procedural knowledge of our reactive agent on the scenario level is similar to the procedural knowledge of the active agent with only one distinction: the agent must know what particular answer must be returned to the active agent as a response to his current question.

Procedural knowledge of an active agent on the level of a dialogue step is its ability to generate a question, which is relevant to the current step (ability to generate subject and prerequisite of the question); and procedural knowledge of a reactive agent on the level of a dialogue step is the ability to construct an answer in accordance to a given subject and prerequisite.

3. Dialogue Knowledge Base

As a procedural-declarative typology of knowledge takes place in question-answering dialogue, it is worth considering the problem of storing declarative and procedural knowledge within a dialogue system architecture. Generally speaking, activity or reactivity is not a fixed attribute of an agent, but rather a role which an agent plays within a dialogue segment. Let us consider a case when these roles are fixed, and a program system simulates the behavior of an active agent. From our point of view, in such a case, the most efficient architecture is an architecture, which is based on the idea of *separate storing of declarative and procedural knowledge*.

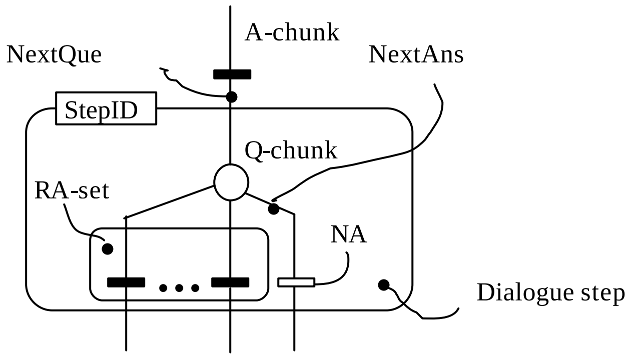
Let declarative knowledge of an active agent (in the form of encoded descriptions of questions needed for the question-answering dialogue in a given domain) be stored in the *memory of questions*, QueMem. Despite the fact that during the dialogue process a certain question can appear in many parts of the dialogue, QueMem keeps only *one copy* of each question. We consider QueMem as a memory with direct access to its elements; and, hence, we need an address to get access to the concrete question.

Let procedural knowledge of an active agent (on the scenario level) be stored in the structure called a *dialogue access method*, DiAM. The dialogue access method keeps a sort of knowledge such as "which question should be next;" and, therefore, is able to transform the current answer of a reactive agent into the QueMem address.

A *dialogue knowledge base*, DiKB, we define as an aggregate of the *memory of questions*, QueMem, and the *dialogue access method*, DiAM. One of the advantages of such a structure of the dialogue knowledge base is that it excludes multiple storing of encoded descriptions of questions. Storing of declarative knowledge of an active agent requires much more computer memory resources than storing of its procedural knowledge because declarative knowledge of an active agent (represented by question's subject elements) might have not only symbolic, but also non-symbolic, representation in the form of graphical and sound files. DiAM operates only with references to active agent's questions and reactive agent's answers; and, therefore, does not require substantial computer memory resources.

As is shown from the definition of DiKB, an active agent does not "compute" the subsequent question but searches it out in QueMem, using DiAM as a method of achieving the goal. Therefore, we can also consider DiAM as a *certain problem-solving method* of an active agent which the agent uses for achieving the goal. question-answering dialogue is a discrete process with a step as its structural and dynamic element. Figure 1 depicts the structure of question-answering dialogue step in Petri net notation and illustrates also the conceptual basis of DiAM.

Dialogue step includes a reference to the question, which corresponds to the step and is designated by Q-chunk position in Figure 1. As different steps of the dialogue can use the same question, then the step should be marked by a unique step identifier – StepID. Transitions in Figure 1 correspond to answers. Denotation of the dialog step includes two sets of answers for a given step: a set expected of answers, or RA-set, and a set of all other answers designated by NA transition. A set of expected answers unites those answers, which in accordance with the dialogue scenario, is expected on a given step, and hence must be recognized by an active agent. Cardinality of this set, therefore, must vary from step to step. A set of all other answers is modeled by a single transition NA because these answers should not be recognized. Thus any answer which is unrecognizable by an active agent on a given step belongs to NA set.



A-chunk: current answer of reactive agent, Q-chunk: reference on subsequent question in QueMem, RA-set: expected (recognizable) set of answers of reactive agent, NA–all other (non-recognizable) answers of reactive agent, NextQue: link between current answer and subsequent question, NextAns: link between current question and subsequent answer, StepID: step identifier.

Figure 1. Graphical illustration of DiAM main concepts.

4. Dialogue Problem Solving Process

The group of ill-formalized problems is vast. Methods of solving these problems are usually based on such a formulation of the problem *which reduces the process of solving the problem to the procedure of search* of the goal state in the problem space. One of the earliest, and probably most well-known, theories in this area is a model called General Problem Solver (GPS) [13, 14].

There is much in common between the question-answering dialogue and a general strategy of searching for the solution within the problem space, implemented in GPS-agent. General search strategy [17] presupposes a step-by-step and cyclically repeated process of constructing a search tree with the following phases: (1) current collection of frontier nodes is generated (at the first step the collection of frontier nodes consists of a single node, which corresponds to the initial state of the problem); (2) every node from the collection of frontier nodes is tested by a goal-test procedure (if result is positive then search is finished--otherwise, the search continues); (3) a node from the collection of frontier nodes which must be expanded next is selected; and (4) the procedure of node expansion is applied to the selected node, and then the search strategy returns to the first phase.

The ability of GPS to find a solution to the problem (in the form of a goal state) from the declarative-procedural typology of knowledge is related to the category of procedural knowledge because the procedural knowledge of dialogue agents is also related to the ability of finding a goal state (in the form of an answer). Therefore, the dialogue process itself can be considered as a general process of solving ill-formalized problems. However, in the case of question-answering dialogue, procedural knowledge is not concentrated in one agent, but rather shared between both dialogue agents.

In the case of question-answering dialogue, the collection of frontier nodes corresponds to the subject of a question, which an active agent presents to the reactive one. An expanded list of properties (or things), which in fact represents the subject of a question, possesses substantial advantages over a classical collection of frontier nodes. The "capacity" of a subject of a question oscillates from one question to another but does not exceed a certain limit, which is determined by a system of focused attention of a human. Consequently in the case of a dialogue this "capacity" is approximately the same and does not depend on the number of steps. On the other hand, in the case of GPS-agent number of nodes (in the collection of frontier nodes) is an increasing function of the depth of the search tree.

In the case of question-answering dialogue, we do not need to apply a goal test (recognition procedure) to all elements of the expanded list, but only to those elements, which passed on to the answer. Therefore, in some sense, a procedural knowledge of a reactive agent carries out the function of "filtration" of the subject. On the other hand, the GPS-agent, as a rule, must apply the goal test to every node (without exception) from the collection of frontier nodes.

One of the most extensive groups of ill-formalized problems is a group of methods for machine/computer teaching. The history of evolution of these methods demonstrates that: (1) all known methods of machine teaching have an interactive nature and presuppose a dialogue; (2) dialogue mainly realizes only a function of interface between the teaching material and the student [16, 20].

The idea of using a dialogue knowledge base for the construction of personalized tutoring systems was tested during the elaboration of a family of lingua-didactical programs [6, 7]. Our experience of elaboration of ready-for-use software in the area of foreign languages self-study allows us to emphasize two aspects related to the dialogue knowledge base. First, design and filling in the knowledge base on both levels (the scenario level and the dialogue step level) is quite natural for the end-user (in our case it is a teacher) and could be done by the end-user himself. Second, as far as the dialogue knowledge base is formed by an expert without any mediator, the dialogue process yielded by this knowledge base reflects not only the method of teaching a particular person, but also the cognitive identity of the expert.

5. Conclusion

Further elaboration of the theory presented in the paper could be evolved in two directions: investigation of applicability of Neisser's cycle of perception [11] as a psychological basis for the dialogue; and investigation of applicability of object-oriented modeling conception for the purpose of specification of a program simulator for the dialogue agents.

We suppose that at least one of the dialogue agents is a human. Therefore, a "good" model of the dialogue must be adequate to the processes of perception and information processing in a human. In the case when a formal model of the dialogue is based on a relevant psychological model, we can expect that the "artificial" dialogue agents will naturally inherit flexibility and universality of human's perceptual and information processing systems. The model of perception proposed by Ulric Neisser in 1976 integrates the "bottom-up" (from the sensory system to the long-term memory) and "top-down" (from the long-term memory to the motor system) processes into a unified and cyclical process.

The dialogue process, in relation to any of the dialogue agents, is very similar to Neisser's cycle of perception; and, therefore, it is rational to investigate the applicability of Neisser's model to the theory of problem-independent dialogue. In the dialogue process a real environment is substituted by an artificial one (formed by the opposite agent); but it is obvious that perception of the environment (real or artificial), and further processing of perceived sensory events, is realized by the same psychological "rules and laws."

Object-oriented modeling technology achieved such a degree of depth and universality that it could be considered as a generic theory of modeling applicable not only to the program systems, but rather for all kinds of systems. We pay special attention to Unified Modeling Language (UML), which transfers Object-oriented conception into a strict and formalized theory [15]. The singularity and attractiveness of UML is in its diagrammatical notation in which diagrams model various aspects of a system and play the role of certain "formulas" of system's structure, behavior, etc. We believe that the expressive power and modeling ability of UML are not less than, for instance, the system of production rules used by Anderson in his "rules of the mind" [1] and that by means of the UML we can construct certain "diagrammatical formulas" of cognitive systems and agents. However, in describing cognitive systems by UML we have one indisputable advantage, UML "diagrammatical formulas" are ready-to-use specifications for a computer program system.

References

[1] Anderson J. R., *Rules of the Mind*, Hillsdate, NJ, Lawrence Erlbaum Associates, 1993.

[2] Anderson J. R., Corbett A. T., Koedinger K. R., and Pelletier R., "Cognitive Tutors: Lessons Learned," *The Journal of the Learning Science*, vol. 4, no. 2, pp. 167-207, 1995.

[3] Belnap D. N. and Steel T. B., *The Logic of Questions and Answers*, New Haven and London, Yale University Press, 1976.

[4] Chemyr A. I., "Dialogue Behavior as a problem solving method and its informal model," Karpina D. and Evdokimova T. (Eds), *Collection of Scientific Works of the Institute of Problems of Modeling in Energetic*, National Ukrainian Academy of Science, Kiev, Ukraine, issue 4, pp. 140-145, 1998.

[5] Chemyr A. I., Horney M. A., and Anderson-Inman L., "SmartText: Using Agents Supporting Personalized Reading Comprehension," *Personal Technologies*, vol. 2, no. 3, pp. 152-161, 1998.

[6] Chimir I. A., Bondar A., Abu-Dawwas A. W., and Chemyr A. I., "Computer Tutor of the Ukrainian Language: Theoretical Basis and Practical Experience," *Scientific Knowledge: Methodology and Technology*, South Ukrainian State Pedagogical University, Ukrainia, no. 1-2, 1998.

[7] Chimir I. A., Popov P. S., and Kharkovskaya M. S., "Dialogue Processor for the Generation of Teaching Courses," *in Proceedings of The International Conference (INFO'89)*, Minsk, USSR, vol. 1, part 2, pp. 957-962, 1989.

[8] Cohen N. J. and Squire L. R. "Preserved Learning and Retention of Pattern-Analyzing Skills in Amnesia Using Perceptual Learning," *Cortex*, vol. 17, pp. 273-278.

[9] Haberlandt K., *Cognitive Psychology*, Allyn & Bacon, Needham Heights, MA, 1997.

[10] Minsky M., "A Framework for Representing Knowledge," in Winston P. H. (Ed), *The Psychology of Computer Vision*, New York, McGraw-Hill, 1975.

[11] Neisser U., *Cognition and Reality Principles and Implications of Cognitive Psychology*, W. H. Freeman and Company, San Francisco, 1976.

[12] Newell A., *Unified Theories of Cognition*, Harvard University Press, 1994.

[13] Newell A. and Simon H. A., "GPS, a Program that Simulates Human Thought," in Feigenbaum E. A. & Feldman J. (Eds), *Computers and Thought.* New York, McGraw-Hill, 1963.

[14] Newell A., Shaw J. C., and Simon H. A., "Elements of a Theory of Human Problem Solving," *Psychological Review*, vol. 65, pp.151-166, 1958.

[15] Page-Jones M., *Fundamentals of Object-Oriented Design in UML*, Addison-Wesley, 2000.

[16] Richmond W. K., *Teachers and Machines: An Introduction to the Theory and Practice of Programmed Learning*, Collins, London & Glasgow, 1965.

[17] Russell S. and Norvig P., *Artificial Intelligence: A Modern Approach*, Prentice Hall, New Jersey, 1995.

[18] Ryle G., *The Concept of Mind*, London, Hutchinson, 1949.

[19] Verlan A. and Chimir I. A., "Systems with Embedded Intelligence Based on the Architecture of a Dialogue Machine," *Electronic Modeling*, Kiev, Ukraine, vol. 23, no. 1, pp. 75-83, 2001.

[20] Wenger E., *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*, Morgan Kaufmann Publishers, California, 1987.

Igor Chimir is a professor of computer science at the Cherkassy Academy of Management, Ukraine. He has been involved in cognitive science research for more than 15 years and has particular interests in the subject-independent theory of computer and human dialogue, intelligent tutoring systems, and in object-oriented modeling of cognitive processes. Dr. Chimir has published more than 90 technical papers and is the author of several textbooks. His current activities focus on investigating the applicability of modern software engineering approaches to the area of cognitive psychology.

Waheeb Abu-Dawwas is an assistant professor at the Department of Computer Information Systems, Al-Zaytoonah University, Jordan. He obtained his PhD from the Ukrainian National Academy of Science, Ukraine, in 1999. His research interests include dialogue process modeling, and the development of personalized computer educational systems. Dr. Abu-Dawwas has published many original contributions in the field of dialogue-oriented tutoring systems, personalized tutoring systems, and in cognitive systems.