Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. IV (2009), No. 1, pp. 17-26

Artificial Intelligence + Distributed Systems = Agents

Ioan Dzitac, Boldur E. Bărbat

Ioan Dzitac
"Aurel Vlaicu" University of Arad, Faculty of Exact Sciences, Department of Mathematics-Informatics
Str. Elena Dragoi, Nr. 2, Complex Universitar M (Micalaca, zona III), Arad, Romania
E-mail: idzitac@gmail.com

Boldur E. Bărbat
"Lucian Blaga" University of Sibiu, "Hermann Oberth" Faculty of Engineering, Department of Research
10, Victoriei Bd, Sibiu, 550024, România
E-mail: bbarbat@gmail.com

Every established religion was once a heresy
Henry Buckle - "Essays"

Abstract: The connection with Wirth's book goes beyond the title, albeit confining the area to modern Artificial Intelligence (AI). Whereas thirty years ago, to devise effective programs, it became necessary to enhance the classical algorithmic framework with approaches applied to limited and focused subdomains, in the context of broad-band technology and semantic web, applications - running in open, heterogeneous, dynamic and uncertain environments-current paradigms are not enough, because of the shift from programs to processes. Beside the structure as position paper, to give more weight to some basic assertions, results of recent research are abridged and commented upon in line with new paradigms. Among the conclusions: a) Non-deterministic software is unavoidable; its development entails not just new design principles but new computing paradigms. b) Agent-oriented systems, to be effectual, should merge conventional agent design with approaches employed in advanced distributed systems (where parallelism is intrinsic to the problem, not just a mean to speed up).
Keywords: open, heterogeneous, dynamic and uncertain environments (OHDUE); computer-aided decision-making; nonalgorithmic software; bounded rationality; agent-oriented software engineering (AOSE).

1 Introduction. Adapting to the Time(s) of Change

Three dominant features of the post-industrial society relevant here are:

a) *Growing speed of change* (due to the intense positive feedback entailed by Moore's law outcomes: Internet, broad-band technology, semantic Web, Google, [9] etc.);

b) *Growing complexity* (architectural, cognitive, structural [4, 5, 10]).

c) *Globalization* (expressed in IT context mainly through the modern enterprise paradigms). Berners-Lee - who coined also the term *GGG* (Giant Global Graph) to describe the semantic Web as a new stage of *WWW* - utters it very pointed: "I have gone from using a 300 board connection on one of those telephone couplers to a 3 million board connection, so that is a 10,000 factor. So the technology underneath this has tremendously increased in terms of speed and functionality, and the Web technology had happened on top" [http://dig.csail.mit.edu/, 2007].

Thus, modern IT environments, except for simple applications, move towards open and *heterogeneous* (resources are unalike and their availability is not warranted), *dynamic* (the pace of exogenous and endogenous changes is high) and *uncertain* (both information and its processing rules are revisable, fuzzy, and uncertain). Most situations to be controlled are complex and uncertain, and involve parallel processes. Thus, the applications developed to deal with must be *intelligent* [7] (to manage complexity and uncertainty) AND *distributed* (to handle parallelism) are intrinsically non-deterministic and end-users have to interact with them in manners they are not get used to. Moreover, a second (side-effect) vicious circle comes out from the interaction between difficulty to adapt and the follow-on frustration (for instance, the claims regarding "digital manipulation").

In brief, adapting to the speed of change is mandatory and the target of this paper is to show that it entails adopting modern IT paradigms. The approach is based on the homomorphism of the addition in the paper title to that in the famous book of Wirth [22].

Accordingly, Section 2 presents the *rationale* analysing and supporting the evolution *from programs to agents*, emphasizing the temporal dimension. Deepening the investigation, Section 3 explores the unavoidable *paradigm shift* (here, many and diverse paradigms are x-rayed). Section 4 includes the core: the misleading term "*distributed*" is thoroughly *revisited* because distribution was mostly part of the *solution* and is now recognized as main part of the problem too. Just to give more weight to some assertions in preceding *sections, models, methods,* and *mechanisms* based on the paradigms endorsed before, are abridged after earlier papers and commented upon in Section 5. *Conclusions* and *intentions* for future development close the paper (Section 6).

2 Rationale. From Programs to Agents

Like all changes induced by systems with intense positive feedback, the shift from programs to agents is at once a long way (*conceptually*, since it involves multiple paradigm changes), a swift leap (*historically*, since - as IT beings - algorithms are active, whereas agents are teenagers) and a hard chaotic fight (*epistemically*, since AI as a whole is at the same time feared and ridiculed, overrated and denied). Misinterpretations are eased because - confusing program architecture with code structure - some professionals still consider that x-rayed programs are nothing more than implemented algorithms, since in binary form data and instructions are indiscernible.

Moreover, in the case of AI, the argument about the difference between programs and agents is complicated by two conceptual inflations due to overstated advertising:

a) to many programs are labelled as intelligent;

b) most such "intelligent" programs are renamed as agents (with or without minor reshaping). Thus, the notion of agent (and the very metaphor behind it) are blurred.

Object orientation - as almost one and only software engineering paradigm - adds two more hindrances to:

a) already in the 1990s, real-time programming showed that even the conceptual equivalence "*program = object*" exposed intellectual difficulties (e.g., objects like "mutex", "event", or "exception", state "start", method "execute");

b) almost as a corollary, by transitivity, it is commonsensical that the equivalence "*agent = object*" is awful, since nobody could be happy to be considered "intelligent like an object".

Besides, objects have - at most - a very primitive temporal dimension. Tough, this dimension is fundamental because no software entity lacking it could be able to (inter)act in the e-world with other

entities, neither artificial (its peers), nor natural (its human end-users). Tellingly enough, time entered the software universe via data structures like dynamic data (e.g., lists in *Pascal*) or data types for concurrency (from monitors in *Concurrent Pascal* to tasks in *Ada*). Likewise, in AI, a genuine temporal dimension was entailed by distributed systems (to handle the parallelism involved - albeit architectural inefficiency due to approaches based on "light" multithreading and time-sharing). Thus, the homomorphism suggested in the paper title is even deeper than supposed at first.

On the other hand - at least in countries similar to Romania - IT curricula are obviously lagging behind the state of the art. Aspects relevant for this paper are:

a) AI syllabi show a long-lasting flavour of "GOFAI" (*G*ood *O*ld-*F*ashioned *AI*).

b) Even when rigid planning-based AI is replaced by "BIC" (Biologically Inspired Computing), the focus is not on *modelling* - inspired from biologic (sub-symbolic) paradigms -, but merely on *simulating* biologic behaviour.

c) Software engineering syllabi are entirely object-oriented; thus, agents - if considered - are designed as objects (e.g., using JADE).

d) Distributed systems are approached in still more conventional manner: the only concern is to boost speed, not to reflect real-world parallelism.

Hence the rationale is threefold, depending on the perspective:

a) *AI* (intelligent software must be *process*-based, not *program*-based);

b) *Software engineering* (intelligent applications should be agent-oriented, not object-oriented);

c) *IT curricula* (AI and Distributed systems should be merged based on agent-orientation).

3 Paradigm Shift. What is Certain?

Here "paradigm" means: "thought pattern in any scientific discipline or other epistemological context. The Merriam-Webster dictionary defines broadly: a philosophical or theoretical framework of any kind. [...] Perhaps the greatest barrier to a paradigm shift, in some cases, is the reality of paradigm paralysis, the inability or refusal to see beyond the current models of thinking" [http://en.wikipedia.org/wiki/Paradigm#Paradigm_shifts].

A lot of fundamental scientific concepts - inside and outside IT - changed dramatically their intension since IT begun to be the dominant "Novum Organon" of post-industrial technological development. They have been investigated from agent-oriented perspective in [4, 5] and lately in [6] where the manifold paradigmatic shift they contributed to engender was labelled "From Kelvin to Zadeh" (Figure 1) because the focus was on *precision* (unnatural in real world, hence inadequate in software) and the shift described there referred to swap from the conventional "Computing with Numbers" (based mainly on measurements) to the modern "Computing with Words" (based mainly on perceptions).

In addition, Figure 1 tries to remind - or at least to suggest - that:

- *Moore's Law* epitomised as feedback loop - in the way usual in automation and electronics - emphasises not only the growing psychological difficulty to adapt to an unprecedented speed of change but also its known side effects: instability, distortion, complexity (mainly, cognitive), frustration (e.g., the irritation about "digital manipulation").

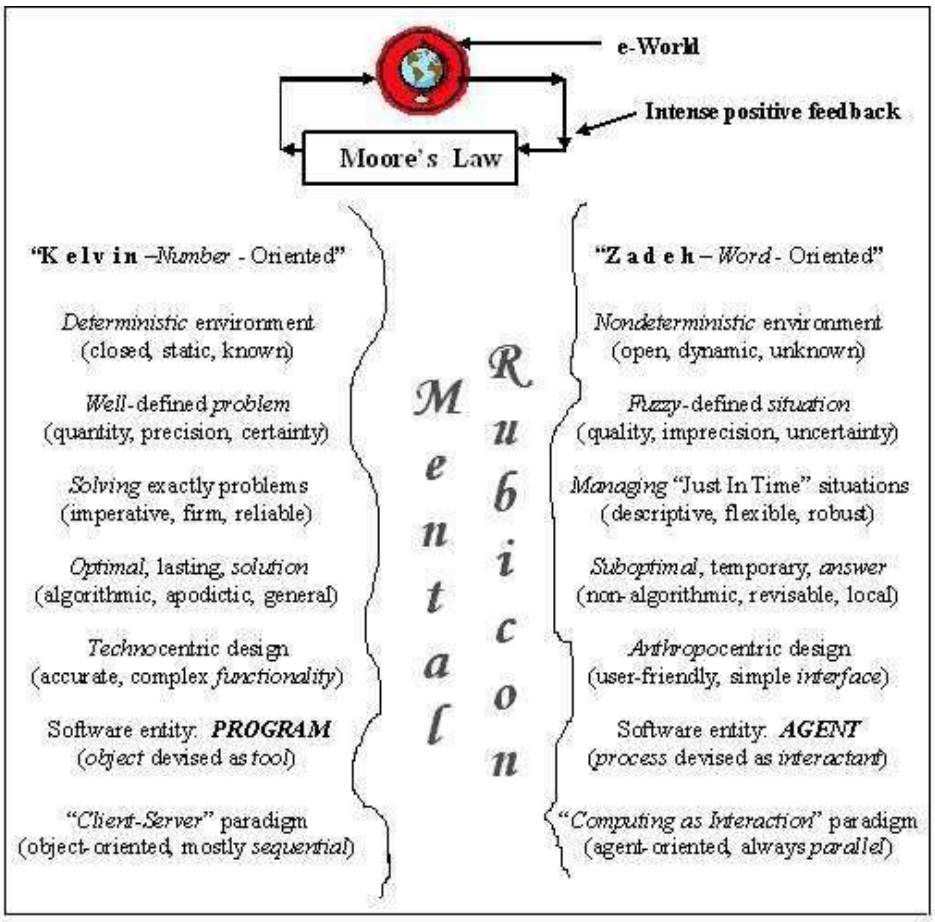

Figure 1: Manifold paradigm shift (adapted and extended from [6])

- "*Manifold*" as "Diverse": In line with common usage, "paradigm" regards also specific areas within a discipline (e.g., "programming paradigm", mainly when it "leverages a computer language's power" [21]). In Figure 1 the paradigmatic level lowers from a wide-ranging one to a narrow, specialised one usual in software engineering. Thus, the "Kelvin paradigm" suggests that IT must stay firmly based on mathematical precision, while the "Zadeh paradigm" replays that it should shift towards semiotic flexibility [23]. On the other hand, shifting from "Client-Server", to "Computing as Interaction" [1, 24]) is focused - not involving (im)precision, (non)determinism, (un)certainty, etc. (Besides, another dimension of reason diversity was recently shown: the ethnographical one [17].)

- "*Manifold*" as "*Many*": After redressing the balance (i.e., accepting all kinds of paradigms, not just GOFAI), AI was inundated by sub-symbolic paradigms; among the best known are artificial neural networks (ANN) and genetic algorithms (GA). The most nihilist and powerful, i.e. the ethological paradigm (based on the physical-grounding hypothesis [4]), seems to be still in vogue for developing agents (above all since the agent is accredited as *process* by a formal standard [12]).

- "*Manifold*" as "*Non-exhaustive*": to save space some important paradigm shifts having rather philosophical/epistemological nature are skipped over. Just one example: humans can communicate only among themselves or with machines too? Here, the problem could be circumvented replacing "*communication*" by "*interaction*" but the epistemological facet is still there: it has to be admitted (or rejected) that, in the framework of "computing as interaction", their connotations are

very similar. Perhaps, the definition given by Sieckenius de Souza could help: "*Communication* is the process through which, for a variety of purposes, sign producers (i.e., signification system users in this specific role) choose to express intended meanings by exploring the possibilities of existing signification systems or, occasionally, by resorting to non-systematized signs, which they invent or use in unpredicted ways" [19].

- Bounded Rationality. The term is used as defined, explained and endorsed in [20, 18, 14]. Since it is a very rich concept, often applied as principle, there are many "models of bounded rationality" [20] - including a Nobel Prize winner one ("psychology for behavioural economics" [16]). It entails that almost any human undertaking - to be effective as regards the time required - must be imperfect. In AI context, perfection suggests the ideal of mathematicians - and conventional software developers too - to achieve algorithm-based optimisation. Just one unquestionable example: "The problem is neither to admit that for any medical act (and for even stronger reasons as regards nursing) "just in time" is a sine qua non condition, nor that bounded rationality is the only practical means to achieve it [13]. Nevertheless, there is a double hindrance, due to a yet prevalent mentality:

  a) therapeutic decision-making is an exclusively human attribute;

  b) non-algorithmic software is - if not nonsensical - applicable at most to toy problems" [5].

  Hence, bounded rationality explodes in a fascicle of interrelated, versatile, and highly application-dependent features. Examples: learning or negotiation strategies; most features meant to reduce complexity (like the "zero-overhead rule" in generative programming).

- Time enters the picture threefold:

  a) From left to right, it represent the very essence of paradigm shift in the sense of Kuhn (for some conservatives unable to adapt the time to accomplish the shift can attain infinity; on the other hand, children have no problem to "communicate" with their artificial playmates).

  b) Explicitly, through the need to manage "Just In Time" rapidly changing situations.

  c) Implicitly, by handling real-world parallelism. Here, parallelism is intrinsic to any interaction (even between an interface agent and its owner), because interactants coexist in time. Of course, distribution adds new facets to an already complex temporal dimension.

In short, to be effectual in e-World both users and software developers have to pass the mental Rubicon separating *programs* from *agents*. Indeed, agents are here to stay (even if not yet very intelligent); "affective computing", "semantic web", "ambient intelligence" and so on, are more than slogans (they are recognized as main IT development directions by EU-promoted acts [1]).

Thus, *"What is certain?"* in the section title goes beyond the necessary enquiry of uncertainty, questioning the very essence of nowadays agent-oriented application development. For instance, even when blending vigorous well established paradigms, research trends are in line with the suggestions in the right part of Figure 1: "Interpretability is considered to be the main advantage of fuzzy systems over alternatives like neural networks, statistical models, etc. [...] In the recent years, research has started to focus on the trade-off between interpretability and accuracy [...]. Analysis of the model interpretability and comprehensibility is always convenient, and it is a necessity when accuracy is not a model feature" [15].

4 "Distributed" Revisited. A Solution becomes Problem

To distribute means "To divide and dispense in portions" implying a previous entirety able to be divided [http://www.thefreedictionary.com/Distributed]. Paradoxically, the epistemic trouble with dis-

tributed systems is that they are conceived, designed and implemented corresponding to the common connotation of "Distributed" [11, 8]. Thus, "In distributed computing a program is split up into parts that run simultaneously on multiple computers communicating over a network. [...] The main goal of a distributed computing system is to connect users and resources in a transparent, open, and scalable way. [...] Distributed computing implements a kind of concurrency. It interrelates tightly with concurrent programming so much that they are sometimes not taught as distinct subjects [...] If not planned properly, a distributed system can decrease the overall reliability" [http:// en.wikipedia.org/wiki/Distributed_computing].

"Distribution", as "the act of distributing or the condition of being distributed; apportionment" [http://www.thefreedictionary.com/distribution], is related to resource management (if something is plenty, no need to apportion).

In short - and maybe oversimplified - there are four kinds of circumstances where, within the IT treatment of the case, *distribution* - in its conventional meaning - can help (inside parentheses are examples) [4]:

a) *In space*. Spatial distribution is the oldest and most familiar type of resource *apportionment* (equipment components, process phases, networking, credit-card terminals).

b) *In time*. Time-sharing preceded its name (learning in schools, delegating authority, or reading on a ride are much older than UNIX-like operating systems or parallel buses).

c) *In organization*. Any organism is based on distributed order (human body, company, state, flower). For virtual enterprises it becomes a major raison d'étre.

d) *In problem solving*. "Divide et impera" was always a foremost strategy to fight complexity, chiefly cognitive one (most reductionist theories, most methodologies - from Euclid's algorithm to structured programming).

The difficulties begun when "distributed", and "parallel" were perceived as quasi-synonymic in the syntagm "distributed computing". Moreover, epistemic confusion escalates when other debatable (semantically antinomic) concepts generated even more doubtful pair of antonyms: "sequential"/"simultaneous" (instead of "parallel"), "holistic"/"analytical" (instead of "reductionist"). A relevant case is related to the basic metaphor of ANN: despite the same distributed neural network structure in both brain hemispheres, an antinomic pair of functions is stated as "linear algorithmic processing" vs. "holistical algorithmic processing" [http://en.wikipedia.org/wiki/Cerebral_hemispheres].

Here the reluctance to accept the paradigmatic shift that processing could be also non-algorithmic (even in the left brain) generated a quasi-pleonasm (could a step-by-step procedure be nonlinear?) and a quasi-contradiction (could a deterministic procedure be holistical?). As regards *processing*, the real opposition is "sequential"/"parallel", where parallelism involves distribution. For instance, "as regards the *learning process* as such - prefixed with "e-" or not - the viewpoint is that human learning is best described by the information-processing approach in cognitive psychology, in line with the ideas endorsed in [2]: "Most modern information-processing theories are "learning-by-doing" theories which imply that learning would occur best with a combination of abstract instruction and concrete illustrations". Learning should be considered - in both humans and agents -as a process where most effectiveness is reached through a blend of symbolic ("left-hemisphere"-like) and subsymbolic ("right-hemisphere"-like) modi operandi" [5].

Though, confusion become delusion when "concurrency" and "distribution" were perceived as conceptually close enough that agent-oriented applications - concurrent *par excellence* - could be effective if they are designed using mechanisms conceived (and used successfully) for distributed systems. Since that is a central claim of this paper, it must be elaborated a bit.

When designing distributed systems - examples above show it clearly - distribution was mostly part of the *solution* not part of the *problem*. Indeed, in most cases, the problem was a whole and, *iff* it

such a problem can be split into subproblems, the entirety is disaggregated, the subproblems are solved and, finally, the partial solutions are re-aggregated. At the programming level they run in parallel and, when accessing shared resources, need *mutual exclusion*. On the contrary, applications devised in line with the "Computing as Interaction" paradigm - above all agent-based applications - entail at least two interactants (the interface agent and its owner), evolving in *parallel*, *autonomously* but not *independently* (since they interact as in any normal communication process: inform, wait, interrupt each other, etc.). Briefly, they are *concurrent* processing threads; their programming entails *multithreading*. The crucial software engineering problem is that, while an API (application programming interface) able to support multithreading covers all requirements for mutual exclusion, the opposite is true just in very simple cases. Worse, in most cases, designing concurrent applications with API intended for distributed systems results in severe ineffectiveness. Hence, acknowledging the difference between distribution and concurrency is paramount not just at epistemic level, but at engineering level too. A relevant step into diminishing confusions was the different name given in *C#* to a instruction existing in *Java*, without changing its semantics: *"Synchronize"* is now called *"Lock"* - expressing what it really does (tellingly, almost the opposite of what its previous name claimed to do, since - to preserve coherence - it reduces parallelism).

5 Models, Methods, Mechanisms

Beside the structure as position paper, to give more weight to the assertions regarding the paradigm shifts symbolised in Figure 1, results of recent research (in line with the paradigms endorsed above) are abridged and commented upon. All software entities mentioned below are *models, methods,* and *mechanisms* but they are abridged without grouping them corresponding to their kind, since they are commented upon in software engineering papers, as well as in [6, 5] and papers referred to there.

There are three categories of mechanisms developed for affordable non-algorithmic agent-based applications in OHDUE:

A) Innovative mechanisms dedicated to "Computer-Aided *x*", where *x* stays for almost any intellectual activity, within the software engineering toolbox AGORITHM (*AG*ent-*OR*iented *I*nteractive *T*ime-sensitive *H*euristic *M*echanisms). So far they are implemented or in earlier development stages, but only for solving toy problems or even in simple experimental models, for *x* = Decision, Learning, Semiosis.

B) Existing mechanisms employed in previous research (before 2005, mainly in experimental models for captologic virtual therapists, carried out as pseudoavatars). Their structure is based on common API functions callable from a customary Java development environment.

C) Conceptualized within the framework of some PhD theses in preparation. To increase paradigmatic relevance they are ordered in relation to Figure 1, that is focusing on the missing "L" in the toolbox name. (Of course, a bijection is out of question.):

- *Decision-Making with Future Contingents. DOMINO* (*D*ecision-*O*riented *M*echanism for "*I*F" as *N*on-deterministic *O*perator). Developed primarily for decision making (typical application: managing overbooking) it is meant to deal with undecidability due to any kind of future contingents. It is a "three-output IF" where the semantics of the third value is a blend of a Łukasiewicz "i" interpreted as "unknowable" and a Kleene *"u"* interpreted as "temporary lack of knowledge". (In fact, the semantics of "Undecidable" is re'fined to "Undecidable in the time span given".)
- *Analog Input*. Scrollbar input is proposed for all kind of data: uncertain knowledge, intrinsically fuzzy, roughly estimated, etc.

- *Dynamic priorities*. Are applied for:
  a) fine-tuning agent priorities (mainly the features of Multi-Agent Systems) to manage situations "Just In Time";
  b) fading out retention in "thick time";
  c) boosting response of exception handlers.
- *Exception-Driven Reactivity*. Prompt response to asynchronous must be mostly stimulus-driven because interaction between basically reactive entities. To respond promptly, interrupts are reflected asynchronously in exceptions with dynamic propagation.
- *Antientropic Self-Cloning.* Developed to implement "strange loops" (via Gödelian self-reference) as first step of investigating agent self-awareness, it means spawning an agent identical to itself, preserving self-representation (its *"I"*), but with an enriched world model (via Lamarckian evolution).
- *Flexible Cloning.* To reach efficient polymorphism the copies are purposely imperfect, spawning an agent quasi-identical to its parent; differences between clones become extensive only after recurring cloning (a clone is just a "slightly altered alter ego").

Unfortunately, the mechanisms listed above have - beside lacking validation *in vivo* (some of them not even in *ovo*) - a double vulnerability: they are either incremental as regards the "Kelvin way of thinking" or too loosely linked to new paradigms. Thus, what is their relevance? To break the vicious circle - since there is no "methodology for paradigm shift" - to leave behind the 3^{rd} Order Ignorance [3], software should be considered "not a product, but rather a medium for the storage of knowledge. [...] The other knowledge storage media being, in historical order: DNA, brains, hardware, and books. [...] Software development is not a product-producing activity, it is a knowledge-acquiring activity." (That is neither fact, nor proof, it is just expectation.)

6 Conclusions and Intentions

a) Non-deterministic software is unavoidable; its development entails not just new design principles but new computing paradigms.

b) Agent-oriented systems, to be effectual, should merge conventional agent design with approaches employed in advanced distributed systems (where parallelism is intrinsic to the problem, not just a mean to speed up).

c) the AGORITHM toolbox - still not sufficient as technological infrastructure for agent-oriented software - is a good framework to go ahead.

d) The paradigm shift "from Kelvin to Zadeh" becomes urgent to keep pace with a rapidly changing e-world.

e) Almost as corollary, these paradigm shifts entails also attitudinal ones: shifting from *multi-*, through *inter-*, towards genuine *trans*-disciplinarity.

As regards the prospects of non-algorithmic agent-oriented software, short-term intentions include enhancing the AGORITHM toolbox with two mini-ontologies:

a) *Dynamic: I* (agent), *You* (master), and *Rest of the world*.

b) *Visual*: Visual rules should simulate *"the arrow of time"*.

Acknowledgement. This work was partially supported by the Ministry of Education and Research through contract No. 12 - 092/2007.

Bibliography

[1] AgentLink III. *Agent based computing. AgentLink Roadmap: Overview and Consultation Report.* University of Southampton. http://www.agentlink.org/roadmap/al3rm.pdf, 2005.

[2] J.R. Anderson, L.M. Reder, H.A. Simon. *Applications and Misapplications of Cognitive Psychology to Mathematics Education.* Texas Educational Review, 2000.

[3] P.G. Armour, *The Five Orders of Ignorance.* Comm. ACM, 43 (10), 17-20, 2000.

[4] B.E. Bărbat, *Agent-Oriented Intelligent Systems.* Romanian Academy Publ. House, Bucharest, 2002 (in Romanian, "Grigore Moisil" Prize of the Romanian Academy).

[5] B.E. Bărbat, E-Maieutics. Rationale and Approach. *International Journal of Computers, Communications & Control*, 3, Supplement: Suppl. S, 40-54, 2008.

[6] B.E. Bărbat, Natural time for artificial agents. *Abstracts of ICCCC Papers*, Băile Felix, May 15-17, 27-27, 2008.(Invited paper.)

[7] I. Dziţac, *Artificial Intelligence.* Ed. House "Aurel Vlaicu" University, 2008 (in Romanian).

[8] I. Dziţac, *Parallel and Distributed Methods for Algebraic Systems Resolution*, CCC Publications, Agora University Publishing House, 2006. (in Romanian).

[9] I. Dziţac, ICCCC 2008 & EWNLC 2008 Celebrates Bardeen's Centenary and Welcomes Professor Zadeh, *International Journal of Computers Communications & Control*, 3 (Suppl. S):16-25, 2008.

[10] I. Dziţac, I. Moisil, Advanced AI Techniques for Web Mining, *Proc. of 10th WSEAS International Conference on Mathematical Methods, Computational Techniques and Intelligent Systems* (MAMECTIS '08, Corfu, Greece, 343-346, 2008.

[11] I. Dzitac, G. Moldovan, *Distributed Systems: Information Models*, Agora University Publishing House 2006 (in Romanian).

[12] FIPA TC Agent Management. *FIPA Agent Management Specification*. Standard SC00023K (2004/18/03). http://www.fipa.org/specs/fipa00023/ SC00023K.pdf

[13] G. Gigerenzer, A. Edwards. Simple tools for understanding risks: from innumeracy to insight. *British Medical Journal*, 327, 741-744, 2003.

[14] G. Gigerenzer, R. Selten. *Bounded Rationality*. MIT Press, Cambridge, 2002.

[15] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evol. Intel.*, 1, 27-46, 2008.

[16] D. Kahneman, *Maps of Bounded Rationality: Psychology for Behavioral Economics.* Lecture (when receiving Nobel Prize; revised version). Stockholm, Nobel Foundation, 2002.

[17] E. Livingston, *Ethnographies of Reason.* Ashgate, Aldershot, UK, 2008.

[18] A. Rubinstein, *Modeling Bounded Rationality*. MIT Press, Cambridge, 1998.

[19] C. Sieckenius de Souza, *The Semiotic Engineering of Human-Computer Interaction.* The MIT Press, 2005.

[20] H.A. Simon, *Models of Bounded Rationality.* MIT Press, Cambridge, 1997.

[21] D. Spinellis, *Rational Metaprogramming.* IEEE Software, 25, 1, 78-79, Jan/Feb, 2008.

[22] N. Wirth, *Algorithms + Data Structures = Programs.* Prentice Hall, 1978.

[23] L.A. Zadeh, D. Tufis, F.G. Filip, I. Dzitac, (Eds.), *From Natural Language to Soft Computing: New Paradigms in Artificial Intelligence*, Romanian Academy Publishing House, 2008.

[24] F. Zambonelli, A. Omicini. Challenges and Research Directions in Agent-Oriented Software Engineering. *Autonomous Agents and Multi-Agent Systems*, 9, 253-283, Kluwer Academic Publishers, 2004.

Ioan Dzitac received M. Sc. in Mathematics (1977) and Ph.D in Information Sc. (2002) from "Babes-Bolyai" University of Cluj- Napoca. His current research interests include different aspects of Artificial Intelligence and Parallel and Distributed Computing. He has edited 6 conference proceedings, published 16 books and more than 50 scientific papers in journals and conferences proceedings. He was member of the Program Committee of more than 30 international conferences.

Boldur E. Bărbat M.Sc. in Electronic Engineering, postgraduate specialising in Programming, Ph.D. in Digital Computers ("Politehnica" University Bucharest). He is with "Lucian Blaga" University Sibiu from 1997, (full professor, Faculty of Engineering, Faculty of Sciences) and "Politehnica" University Timişoara from 2005 (Faculty of Automation and Computers, advisor for doctoral studies in Computer Science). 25 books (a monograph received the Romanian Academy IT Prize, 2002). About 50 papers in English from 2001. Current research interests: Time in Artificial Intelligence; Nonalgorithmic decision support systems; Computer-Aided Semiosis; Self-awareness of bodiless agents; Agent-Oriented Software Engineering; Logics for agents; Emergence in Agent-Based Systems.