# Hybrid Intelligent Systems in Ubiquitous Computing

Dr. Andrey V. Gavrilov
Novosibirsk State Technical University, Russia
Andr_gavrilov@yahoo.com

**ABSTRACT.**
In this chapter hybrid approach to development of intelligent systems is applied to ubiquitous computing systems, in particular, to smart environment. Different classifications of Hybrid Intelligent Systems (HIS) are looking and two examples of hybrid approach for smart environment are suggested: framework based on expert system and neural network for programming of behavior of smart objects and paradigm of context-based programming-learning of behavior of intelligent agent. Besides this chapter offers an attempt to systematize concepts for development of HIS as any introduction to methodology for development of HIS is suggested. Author hopes that this chapter will be useful for researchers and developers to better understand challenges in development of ambient intelligence and possible ways to overcome them.

**Keywords:** Neural Networks, Knowledge Models, Fuzzy Systems, Knowledge-Based Software, Cybernetics, Hybrid Intelligent Systems, Ubiquitous Computing Systems, Artificial Intelligence, Ambient Intelligence

## INTRODUCTION

Now ubiquitous computing, ambient intelligence and smart cooperative objects are viewed as a major paradigms shift from conventional desktop application development. This view is enabled through the use of diverse hardware (sensors, user devices, computing infrastructure etc.) and software, anticipating user needs and acting on their behalf in a proactive manner [Weiser, 1991; Satyanarayanan, 2001]. This diversity of hardware and software information increases the degree of heterogeneity.

In order to realize such ubiquitous computing environment, three technology areas are required:

1) Sensing technology where information on user and surrounding environment are perceived and collected,
2) Context aware computing [Schilit, Adams & Want, 1994; Baldauf & Dustdar, 2004] technology where such information are processed and properly presented to users as different services,
3) Wireless network technologies [Mahalik, 2007] where information are collected from sensors and distributed to customers – services and users.

One of most perspective technologies for sensing and perception is neural networks. We may pick out following main features of ubiquitous computing systems (UCS):

1) distribution of obtaining and processing of sensor information,
2) variety of information needed processing,
3) necessity of learning during interaction with environment, in particular, in respect to existing of unexpected events and objects needed for including into processing,
4) key role of different kinds of human-machine interaction,
5) high requirements to security,
6) data processing in real time,
7) wide usage of embedded processing units.

There are following tasks for neural networks in development of ubiquitous computing systems:

1) perception, i.e. recognition of objects and changes in environment, in particular, invariant recognition of moving objects, e.g. recognition of gesture, position and emotions of human beings,
2) clustering and recognition of events and scenarios (sequence of events in time),
3) prediction of future events and situations,
4) indoor localization of mobile devices and continues mapping,
5) reactive behavior based managing of actions,
6) speech recognition.

From above we can formulate following requirements to neural networks for UCS:

1) Relatively fast processing of information in both learning and recalling,
2) Incremental learning, i.e. availability to perceive new information without loss of old knowledge,
3) Availability of easy extraction of structure from learnt neural network for building of symbolic knowledge for usage in machine-human interaction and planning.

On the other hand context awareness usually is implemented by symbolic based reasoning and knowledge-based techniques [Hung, Shehzad, Kiani, Riaz, Ngoc & Lee, 2004]. Besides rules based approach is appropriate for human-machine interface for programming of behavior of smart objects [Tarik, Sarcar, Hasn, Huq, Gavrilov, Lee & Lee, 2008] and for any explanation for user.

The following tasks are more relevant to rule based and other symbolic techniques:

1) A prior description of behavior of smart object with respect to perceived objects/situations and context, including managing of dialog with user;
2) Reflex to perceived important situation starting determined behavior;
3) Diagnostics of sensor network;
4) Specific tasks in ubiquitous computing system, for example, dealing with medical diagnostics of patient in healthcare system, or decision making in recommendation systems;
5) Human-computer interaction based on natural language.

Therefore, obviously that hybrid intelligent system (HIS) approach based on combination of neural networks and formalized knowledge is most perspective for implementation of ambient intelligence in ubiquitous computing systems [Gavrilov, 2008, 3].

There are some challenges to be solved for usage of Hybrid Intelligent Systems in UCS:

1) Cooperation between sensing neural networks and reasoning knowledge structures, consisting of forward and feedback connections between them, transformation from one to another,
2) Knowledge acquisition from learnt neural networks,
3) Architecture of distributed hybrid intelligent system in heterogeneous wireless network.

In this chapter author are focusing on first of them, looking different kinds of HIS and suggesting two cases of hybrid intelligent systems in UCS: architecture of hybrid embedded expert system for control of smart objects [Gavrilov, 2008, 2] and context-based programming-learning of smart objects, in particular, mobile robots, using natural language [Gavrilov, 2008, 1; Gavrilov, 2009]. Besides, in this chapter author try to introduce systematic view on methodology for development of hybrid intelligent systems.

## HYBRID APPROACH TO DEVELOPMENT OF INTELLIGENT SYSTEMS

Hybrid approach in wide sense is combination of different paradigms of knowledge representation and reasoning such as rules, semantic nets, frames, fuzzy logics, neural networks, genetic algorithms, swarm intelligence and so on. Alternate and narrower view on one is combination or fusion in one system symbolic knowledge based and associative neural based (connectionist) techniques. Last opinion is more interesting for us and more perspective

because is more constructive and is attempting to simulate processes in natural mind. So in this paper we will talk about this paradigm of hybrid intelligent systems.

Before appearance of hybrid intelligent system paradigm these techniques were investigating separately in classical Artificial Intelligence and neurocybernetics (later Computational Intelligence) and it is continued. Thus in these areas researches obtained many efficient and suitable techniques. To combine these one it is needed to think about interfaces between them. According to features of interfaces different classifications of hybrid intelligent systems are known.

In [Medsker & Bailey, 1992] first classification of HIS was proposed. Note that in this paper authors say about classification of hybrid expert systems but one is possible to extend on other kinds of intelligent systems. They suggested following kinds of hybrid expert systems: stand-alone, transformational, loosely-coupled, tightly-coupled and fully-integrated models. This classification focuses on features of system structure and interaction between different paradigms.

Stand-alone model of HIS consists of independent software components. In this case interaction between them to solve any task is possible only by human.

Transformational model is similar to stand-alone model but in this case we have any automated mechanism of transformation of knowledge base to neural network and vice versa. Usually this mechanism is executed sometimes during development and/or exploitation of system.

Loosely-coupled models comprise the first true form of integrated intelligent systems. In this model components interact between them but this interaction is enough brief and one is executed through file. That allows implementing these components independently providing any simple interface by this file.

Tightly-coupled model is oriented on more close cooperation between components. However, tightly-coupled systems pass information via memory resident data structures rather than external data files.

Fully-integrated model shares data structures and knowledge representations between different paradigms. Communication between them is accomplished via the dual nature (symbolic and neural) of the structures.

In [Goonatilake & Khebbal, 1995] was suggested another classification of hybrid intelligent systems. This classification focuses on connection between functionality and different paradigms and consists of three kinds of HIS: functional-replacing, intercommunicating and polymorphic hybrids.

Function-replacing hybrids address the functional composition of a single intelligent technique. In this hybrid class, a principal function of the given technique is replaced by another intelligent processing technique.

Inter-communicating hybrids are independent, self-contained, intelligent processing modules that exchange information and perform separate functions to generate solutions. These independent modules, which collectively solve the given task, are coordinated by a control mechanism.

Polymorphic hybrids are systems that use a single processing architecture to achieve the functionality of different intelligent processing techniques. The broad motivation for these hybrid systems is realizing multi-functionality within particular computational architectures. These systems can functionally mimic, or emulate, different processing techniques.

Another classification of hybrid architectures was proposed in [Funabashi, Maeda, Morooka, Mori, 1995] focusing on features of process structure for solving any task. In this classification authors describe follow kinds of HIS (Fig. 1): combination, integration, fusion and association.

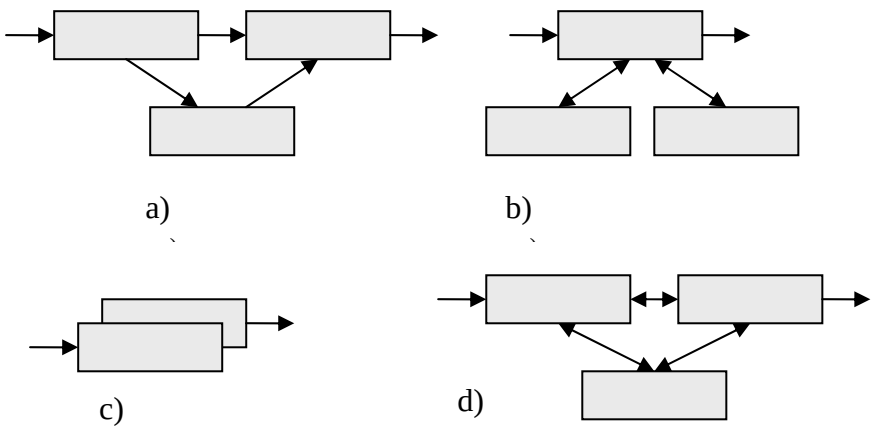


**Figure1. Classification of HIS architectures in [Funabashi, Maeda, Morooka, Mori, 1995]: a) – combination, b) integration, c) fusion, d) association.**

Very similar classification was proposed in [Jacobsen, 1998].

At last, in [McGarry, Wermter, & MacIntyre, 1999] was proposed newest classification consisting of three types of HIS: unified, transformational and modular hybrid systems.

The first group, “unified hybrid systems", consists of those systems that have all processing activities implemented by neural network elements. Such systems may be classified under the unified approach. So far these systems have only limited impact upon real world applications, due to the complexity of implementation, issues of model scalability and rather limited knowledge representation capability. This kind of HIS is basically similar to fully-integrated model in [Medsker & Bailey, 1992], polymorphic HIS in [Goonatilake & Khebbal, 1995] and fusion in [Funabashi, Maeda, Morooka, Mori, 1995].

The second group of systems can transfer a symbolic representation into a neural one and vice versa. It is with the second category, “transformational hybrid systems", that hybrid system begin to demonstrate some unique properties. The most interesting feature is the ability to insert, extract and refine symbolic knowledge within the framework of a neural network system. This kind of HIS is mostly similar to transformational model in [Medsker & Bailey, 1992].

The third category of “modular hybrid systems" covers those hybrid systems that are modular in nature, i.e. they are comprised of several neural networks and rule-based modules which can have different degrees of coupling and integration. An important aspect is that they do not involve any changes regarding the conceptual operation of either the neural network or rule-based components. The vast majority of hybrid systems fall into this category. The main reason is that they are powerful processors of information and are relatively easy to implement. This model is similar to loosely-coupled and tightly-coupled models in [Medsker & Bailey, 1992], inter-communicating HIS in [Goonatilake & Khebbal, 1995] and almost all models in [Funabashi, Maeda, Morooka, Mori, 1995] except fusion model.

At last twenty years hybrid intelligent systems are wide application, in particular, expert systems in different areas [Medsker & Bailey, 1992; Funabashi, Maeda, Morooka, Mori, 1995], control systems for mobile robots [Wermter & Ron Sun 2000; Gavrilov, Gubarev, Jo & Lee, 2004], core technology for development of artificial general intelligence and model of mind [Goertzel, Pennachin, 2007].

Besides structure and protocols features of interfaces between neural based and symbolic components of HIS covered by above classifications, it is needed to overcome challenges dealing with different nature of information processing in these components. These problems are more sophisticated and deep because descend from nature of our mind. For example, how to extract formalizing knowledge from pattern stored in neural network without loss of

sufficient information, how to use formalized knowledge to control of flexible distributed actuator such as hand-like manipulator. This problem is close to such questions as: “What is consciousness and subconsciousness?”, “What roles of them in thinking?”, “What is sufficient information deserving of formalization as knowledge?” and at last “What are basic principles of mind oriented to interaction with unknown and uncertain environment?” (Gavrilov, 2003, 2007). These problems are investigated in different theories of mind and exceed the bounds of this paper.

Below author propose two examples of employment of hybrid approach to ubiquitous computing systems.

## HYBRID EMBEDDED EXPERT SYSTEM FOR CONTROL OF SMART OBJECTS

In [Gavrilov, 2008, 2] author have proposed for development of hybrid intelligent systems for smart environment employment of framework implemented in hybrid expert shell ESWin [Gavrilov & Novickaja, 2001; Gavrilov & Chistyakov, 2005] combining with neural networks (ESWin 2.1 is free downloadable from http://www.insycom.ru). Previously this shell was oriented to development of dialog hybrid expert systems for solving of tasks for diagnostics and recommendations in well structured areas. But now ESWin is improving and expending for building of embedded systems in smart environment and robotics.

Proposed architecture of hybrid intelligent system may be classified as both loosely-coupled and modular with accordance to above classifications.

### Knowledge representation and processing

Knowledge representation in ESWin (fig. 2, fig. 3) is based on frames, fuzzy rules and linguistic variables [Zadeh, 1975]. Frames are used for description of structure of world (area) including properties and hierarchies of entities as frames-classes. Besides frames aim to store current facts about entities in frames-instances. Rules are used for description of solving of tasks, e.g. behavior of any smart object. Linguistic variables are determined inside frames and are used for description of fuzzy properties of objects or situations which may have both quantitative and qualitative values at the same time.

Solving of task is a fuzzy rule based backward chaining using data from frames and perception devices (e.g. user interface or perception subsystem based on neural network). Backward chaining provides execution of any behavior or decision making depending on current set of facts and goal. Goal means what we want to get from this behavior. For example, in simplest case it may be name of behavior.

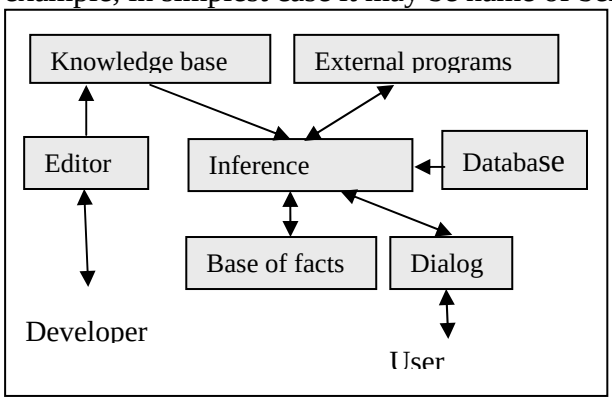


**Figure 2. Architecture of ESWin**

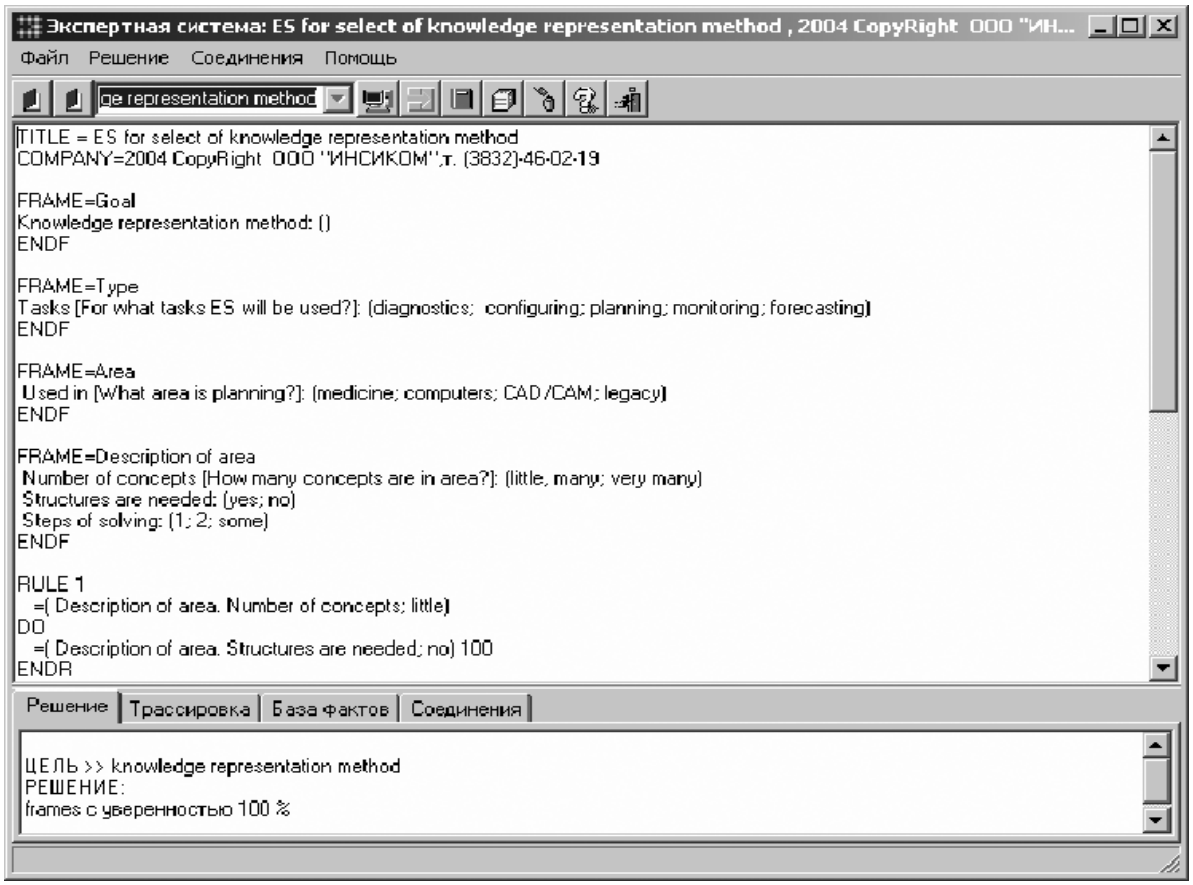


**Figure 3. Screenshot of ESWin's shell for developers**

Let take up ubiquitous service helping to find and operate the objects in room for elderly or handicapped people. This task may be executed by mobile robot too. Assume that we have one or more camera for monitoring of location and state of person. Below you see fragment of knowledge base of this service realized in knowledge representation language of ESWin, modified for smart environment.

```
// This frame describes possible goals for backward
// inference
Frame = Goal
        Find: (Cornflakes box; Book; TV control;
                Medication; Thermometer)
EndF
// These frames are in knowledge base (model of world)
Frame = Cornflakes box
        Parent: Cereal box
EndF
Frame = Cereal box
        Parent: Box
        Action: (Pour out; Fill)
EndF
Frame = Box
        Parent: Object
        Fullness: (Full; Empty)
        Openness: (Opened; Closed)
        Action: (Open; Close)
EndF
```

```
Frame = Object
        Name: ()
        Action: (Take; Put; Find)
        Location: (Into; Visible; Anywhere)
        Into: ()
// Here “lv” is a slot’s type “linguistic variable”
// Linguistic variable’s membership functions are described in another special file
        Where (lv): (Near; Far; Not far; Close by)
        Direction: (Ahead; To left; To right; To back)
        Path: ()
        Color: (Red; Green; Yellow; Blue; White;
                Black)
        Size (lv): (Huge; Large; Small; Tiny)
        Form: (Sphere; Cube; Rectangle; Complex)
EndF
// This and above frames provide generation of frame-fact // from slot Name of Frame-parent
// with Name like value of corresponding slot of Object with inheritance
// of properties of Object
Frame = Name
        Parent: Object
EndF
Frame = Person
        Action: (Turn to right; Turn to left;
                Forward; Stop; Turn back; Go to; Find)
        State: (Stand; Lie; Sit; Walk)
        Direction of view: ()
EndF
// These frames are in base of facts
Frame (instance) = Cornflakes box
        Color: Green
        Location: Into
        Into: Cupboard
EndF
// Facts about visible (recognized) objects
Frame (instance) = Person
        State: Stay
EndF
// Distances to visible objects
Frame (instance) = Cupboard
        Where: 1, 7
        Direction: Ahead
EndF
Frame (instance) = Window
        Where: Not far
        Direction: To left
EndF
// Rules for prompting to find any object being into cupboard
Rule 1
        =(Object.Name; *Object)
        =(*Object.location; Into)
```

```
        =(*Object.into; Object_2)
        =(*Object_2.path; Any)
        =(*Object_2.Direction; Ahead)
        <(*Object_2.Distance; 0,3)
DO
        =(Goal.Find; *Object)
        MS(Action.Ms; Stop, Please take *Object from *Object_2)
EndR
Rule 2
        =(*Object.Into; Object_2)
        =(*Object_2.Direction; Ahead)
        <>(*Object 2.Distance; Close by)
DO
        =(*Object_2.Path; Ahead)
        MS(Action.Ms; Go straight ahead to cupboard)
EndR
```

Note that term **Object* is analog of identifier of variable which replaced by corresponding value from previous steps of interpretation of rules.

In this example some data (values of frame's slots), such as *Direction* and *Distance* to any objects-places (*Cupboard* or *Window*), *Form* of *Object* (*Sphere*, *Cube*, and so on), *Color* (*Green*, *Red* and so on), *State* and so on, may be obtained from sensing subsystem based on one or more neural networks. Any of them may be produced by rules, for example, *Cupboard.Direction* may be got from *Person.Direction of view* by executing of rule:

```
Rule 3
        =(Person.Direction of view; *Object)
DO
        =(*Object.Direction; Ahead)
EndR
```

To get more appropriate language for smart environment the knowledge representation language of ESWin had modified as follow:

1) To provide more compact knowledge base with description of different behaviors, was introduced a concept of variables-references in rules which may assume value (replaced by value similarly as in macro substitution) during interpretation of rules until task is solved (e.g., reference *Object and *Object_2 in our example).

2) To provide capability of solving of task during long time, were introduced two modes of interpretation (inference) – short time (for decision making in one moment) and long time (decision making during monitoring and control like in our example).

3) To provide capability to control of several processes simultaneously (e.g. our fragment of knowledge and any reaction for recognized important situation), multi-task capability was introduced into the interpreter. It means that start of any task (chaining) causes a process. And we have to support interaction between processes, in particular, interruption, waiting and shut down of processes. In above described example the interpretation of Rule 1 may be stopped and waits until condition <(Cupboard.Distance; 0,3) or "distance between person and cupboard < 0,3 meters" is true. It means that process will wait when person (or mobile robot) would come to cupboard. It is obvious that we have to implement any mechanism for interruption and shut down of this process.

4) To provide reactive capabilities is implementing forward chaining (in ESWin 2.1 it is absent). Although it is possible to simulate reactive capability by backward chaining with

multi-task capability and waiting of respect condition. But this simulation of reactive response is not enough suitable, visibility and effective.

**Interaction between inference and neural networks**

To support of interaction between rule-frame based inference and neural networks the following mechanism was proposed in [Gavrilov & Chistyakov, 2005] (fig. 4).

Special data structure SOURCE describes necessary parameters for the definition of interaction of expert shell with other program, in particular, neural network based sensing routine. If knowledge base interpreter will find this special frame in the condition of checking rule during process of inference, it realizes a pause of process and sends an inquiry for the reception of fact from other program. After receipt of fact the process of inference continues. If inquired fact does not enter from the external source in specifying gap of time, the fact is taken as empty (it means absence of the fact) and the inference continues. But if this special frame is used in conclusion rules, the expert system sends a fact to other program. The example of this structure for connection with neural network by socket is:

```
SOURCE = NN1
        Type: Connection
        Port: 1234
        IP: 127.0.0.1
        TimeOut: (10) // time of waiting of response
        Old: (60) // time for aging of fact got from NN1
EndF
```

This structure contains all information needed for interface with neural network with name NN1 by socket. Usage of socket allows deploying modules of this hybrid intelligent system in network of smart environment.

Variables values of which are produced by neural network are described in frame with same name like this structure:

```
FRAME = NN1
        Form: (Cube; Sphere; Rectangle; Complex)
        Color: (Green; Red; Yellow; Blue; White;
                Black)
ENDF
```

If we use forward inference then the neural network is able to produce fact in arbitrary moment of time when it recognizes any situation or object. In both cases of inference (backward or forward) the result of execution of neural network is fact understandable for inference engine.

A dictionary for interface between neural network and inference engine is used. It is employed for coding of words (phrases) contained in frame for interface as vectors for neural network and for decoding of vectors obtained from neural network as words (phrases) constructing facts. In our example described above we must have in dictionary following words: *Form, Cube, Sphere, Complex* (using for recognition of form), *State, Stay, Lie, Sit* (for recognition of state of person), *Object, Cupboard, Cornflakes box, Cup, Bottle, Window* (for recognition of objects).

Note that if any fact checked by rule exists already in base of fact (obtained from neural network earlier) then this fact is used without execution of neural network while difference between current time and time from one's appearance less than *Old*.

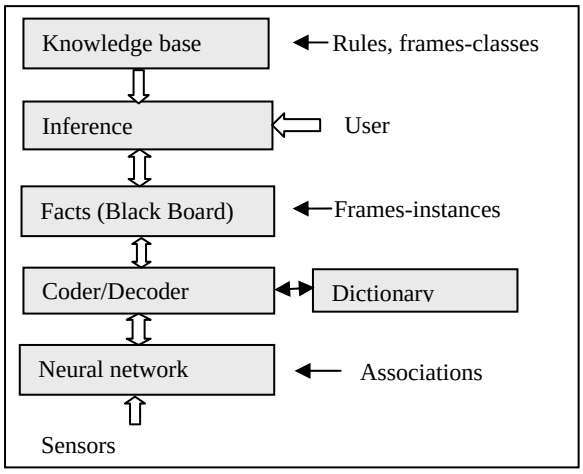


**Figure 4. Layers of information processing**

## CONTEXT BASED PROGRAMMING-LEARNING OF SMART OBJECTS

One of most challenges in development of intelligent robots and other intelligent agents in ubiquitous computing systems is human-computer interface. Two kinds of such interfaces are known, oriented on programming and learning respectively. Concern to robotics a programming is used usually for industrial robots and other technological equipment; learning is more oriented for service and toy robotics. Here author will talk basically about robots but all lower statements can be applied to other kinds of smart objects.

There are many various programming languages for different kinds of intelligent equipment, for industrial robots-manipulators, mobile robots, technological equipment [Wan (Ed.), 2001], [Pembeci & Hager, 2002]. The re-programming of robotic systems is still a difficult, costly, and time consuming operation. In order to increase flexibility, a common approach is to consider the work-cell programming at a high level of abstraction, which enables a description of the sequence of actions at a task-level. A task-level programming environment provides mechanisms to automatically convert high-level task specification into low level code. Task-level programming languages may be procedure oriented [Meynard, 2000] and declarative oriented [Williams, Ingham, Chung & Elliott, 2003; Hudak, Courtney, Nilsson & Peterson, 2002; Vajda & Urbancsek (2003); Wahl & Thomas, 2002; Samaka, 2005] and now we have a tendency to focus on second kind of languages. But in current time basically all programming languages for manufacturing are deterministic and not oriented on usage of learning and fuzzy concepts like in service or military robotics. But it is possible to expect in future reducing of this gap between manufacturing and service robotics.

On the other hand in service and especially domestic robotics most users are naive about computer language and thus cannot personalize robots using standard programming methods. So at last time robot-human interface tends to usage of natural language [Gavrilov, 1988; Lauria, Bugmann, Kyriacou, Bos & Klein, 2001] and, in particular, spoken language [Spiliotopoulos, Androutsopoulos & Spyropoulos, 2001]; Seabra Lopes, Teixeira, Rodrigues, Gomes, Teixeira, Ferreira, Soares, Girão & Sénica, 2003]. Such mobile robot for example must understand such phrases as “Bring me cup a of tea”, “Close the door”, “Switch on the light”, “Where is my favorite book? Give it to me”, “When must I take my medication?”.

Using natural language dialog with mobile robot we need to connect words and phrases with process and results of perception of robot by neural networks. In [Beetz, Kirsch, & Muller, 2004] to solve this problem the extension of robot programming language by introducing of corresponding operators was proposed. But it seams that such approach is not enough perspective.

Here we suggest bio- and psychology-inspired approach to combine programming and learning to perception of robot based on usage of neural networks algorithms and context as results of recognition of concepts obtained by learning during dialog in natural language [Gavrilov, 1988; Gavrilov, 2008; Gavrilov, 2009]. In this approach author do not distinguish learning and programming and combine a declarative (description of context) and procedural knowledge (routines for processing of context implementing elementary behavior) on the one hand, learning in neural networks and ordering of behavior by dialog in natural language on the other hand.

**Proposed paradigm and architecture of intelligent agent**

Our paradigm is based on relationships with other known paradigms and is shown in Fig.5. In Fig.6 you can see architecture of intelligent agent based on this paradigm.

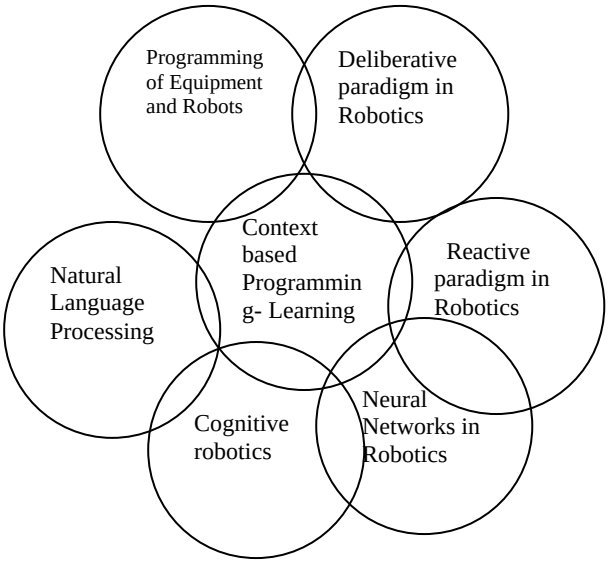


**Figure 5. Relationships between paradigm "Context based Programming-Learning" and other paradigms.**

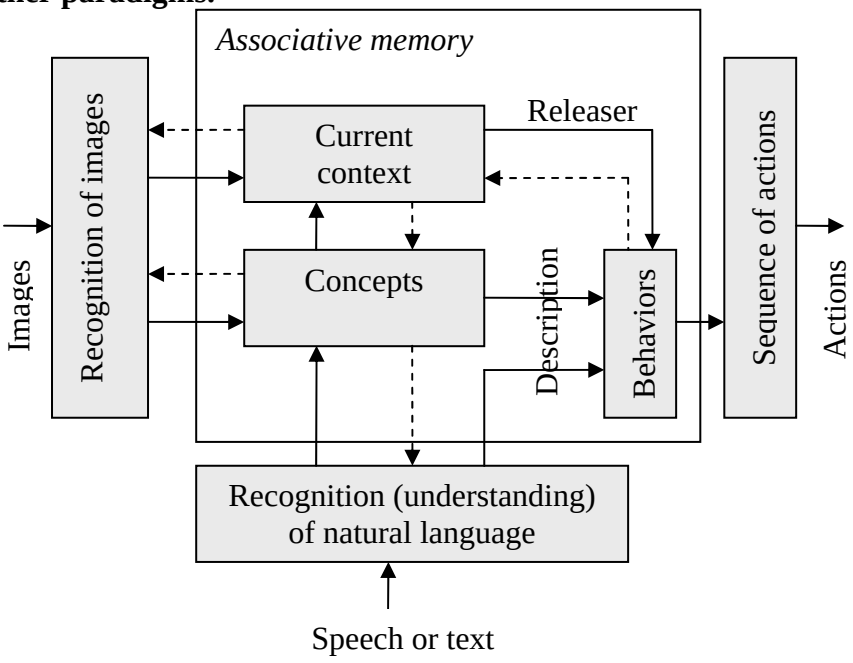


**Figure 6. Intelligent agent functional structure oriented on context based programming-learning.**

Concepts are associations between images (visual and others) and phrases (words) of natural language. In simple case we will name as concept just name of them (phrases). These phrases are using for determination of context in which the robot is perceiving environment (in particular, natural language during dialog) and planning actions. We will name them as *context variables*.

Context is a tree of concepts (context variables) and is recognizing by sensor information and determining by dialog with user. Feedbacks between concepts/context and recognition of images/phrases mean that recognition is controlled by already recognized things.

Dialog with user aims to describe elementary behaviors and conditions for its starting. To start any behavior it is needed recognize corresponding concept-releaser as in reactive paradigm of control system of robot [Murphy, 2000].

Behavior is not influence on just actions but also on context. Moreover, behavior may be not connecting directly with actions in environment. In this case we have just thinking under context variables. And even when behavior is oriented on execution of actions it is possible to block this connection and in this case we have any simulation in mind sequence of actions (e.g., it may be mean as planning).

The associative memory must satisfy to follow requirements:

1) To allow usage of both analog and binary inputs/outputs,
2) To provide incremental learning,
3) To provide storing of chain of concepts (as behavior or scenarios).

The elementary behavior is similar to subroutine and contains sequence of actions adaptable to context variables which may be viewed as parameters of subroutine. Of cause, we need to use some primitives directly connecting with elementary actions and describing parameters of these actions or basic context variables. Connections between these primitives and words of natural language must be a prior knowledge of robot obtaining at development of robot or during a preliminary teaching by specialist. It may be simple language similar to CBLR, proposed by author for context based programming of industrial robots in [Gavrilov, 1988; Gavrilov, 2008]. Feature of this language is absence of different motion primitives. There we have just one motion primitive. All other primitives aim to represent of context variables needed for execution of this motion primitive. To distinguish these primitives and usually using motions or geometric primitives we will call ones below as *output primitives*.

This hybrid architecture may be classified as fully-integrated model or tightly-coupled model in Medsker's classification and fusion or association in taxonomy of Funabashi et al. More concrete kind depends on details of implementation.

**Cross-Modal Incremental Learning and Associative Memory**

Associative memory satisfied to above requirements may be based on hybrid approach and similar to Long-Short Term Memory (LSTM) [Hochreiter & Schmidhuber, 1997] or table based memory proposed in [Gavrilov, Gubarev, Jo & Lee, 2004].

Note that *Concept* in Fig. 6 is more common thing as *context*. Concepts are introducing by dialog subsystem for determination of objects, events, properties and abstraction. But *context* consists of several concepts separated by three rules: 1) preliminary defined concepts (names) are using as names of "parameters" to utilize in elementary behavior; 2) some concepts are using as values of these "parameters"; 3) some concepts (releasers) are using as names of any events causing any behavior.

Dialog subsystem must provide robustness to faults in sentences. For one's implementation may be used approach as proposed in [Gavrilov, 2003] and based on semantic networks combining with neural networks algorithms. This approach is oriented on fuzzy recognition of semantics. Dialog subsystem must provide attend visual recognition subsystem when it is needed to connect word (phrase) with recognizing image. In this case the recognition

subsystem create new cluster with center as current feature vector. And couple of this feature vector and word (phrase) is storing in associative memory for concepts. It may be like result of processing of follow sentence “This is table”. In contrast to this case when system processes sentence “Table is place for dinner” the new cluster is not created and associative memory is used for storing of association between just words (phrases).

Thus associative memory must be able to store associations between both couple of symbolic information and couple word (phrase) and feature vector. Besides concept storing in associative memory must have some tags: basic concept (preliminary defined) or no, name or value, current context or no. And every concept in associative memory must be able to connect in chain with other ones. The order of concepts in chain can be defined by dialog subsystem.

**Output primitives for mobile robot**

Set of output primitives may be selected by different way. One approach to that is determination of enough complex behaviors, such as “Find determined object”, “Go to determined place” and so on. These actions may be named as *motion primitives*. Determined *object* and *place* must be obtained from context as value of corresponding context variable. And these variables may be viewed as another kind of output primitives: *context primitives*. In this case such actions must include inside any enough strong intelligence and ones limit capabilities of mobile robot to learn. Another approach may be based on very simple *motion primitives* such as “act”, which may be just one. If the robot has any manipulator we may use also similar primitive for action of it. In this case may be primitive “act” for manipulator and “move” for mobile platform. If we want to make capability to say anything by robot not only during dialog inside dialog subsystem, we need introduce also at least one motion primitive “say”. All another *output primitives* are *context primitives* and ones define features of execution of primitives “act” and “say”. In other words ones are parameters of subroutine “act”. Examples of robot’s context variables are shown in Table 1

**Table 1.** Examples of context primitives for mobile robot.

| Name of context primitive | Possible value | How this parameter influence on execution of motion primitive |
|---|---|---|
| Object | Name of object | May be used in action “say” |
| Internal state | Good, Bad, Normal | May cause motion to or from *Object* |
| Direction | Left, Right, Forward, Back | May cause corresponding turn depending on *Internal state* |
| Person | Name of person | May be used in action “say” |
| Obstacle distance | Far, Middle, Close | May be used in “act” |
| Obstacle type | Static, Dynamic | May be used in “act” |
| Speed | Low, Normal, High | May be used in “act” |

## INTRODUCTION TO METHODOLOGY FOR DEVELOPMENT OF HYBRID INTELLIGENT SYSTEMS

When we need to build any Hybrid Intelligent System, in particular, for ubiquitous computing system, we must answer follow questions:

- How to select different paradigms of AI,
- How to combine these paradigms,
- How to select interfaces (user and/or others),
- How to connect developed architecture for solving of the tasks with interfaces.

To answer these questions for any concrete task (application) we need:

1) to determine the methods of representation and solving of the task,
2) to perform the forms of input and output data,
3) to select the methods of hybridization.

Process of solving of task may be decomposed by follow ways:

1) Sequence of subtasks,
2) Set of concurrent subtasks, decision is a decision of subtask-winner,
3) Tree of subtasks,
4) Network of subtasks, described by oriented graph.

Real application may use input data of three kinds: Boolean vector, analog vector and symbolic data (in particular, words of natural language). Forms of presentation of these input data may be performed by follow taxonomy:

1) Visual stream;
2) Visual picture;
3) Sound;
4) Voice;
5) Signals from special equipment;
6) Formatted text;
7) Non-structured text.

The output data may be distinguished as:

1) Symbolic data;
2) Numeric data;
3) Signals;
4) Pictures;
5) Animation;
6) Diagrams.

Forms of present of output data may be wrote as:

1) Words of natural language;
2) Sentences of natural language;
3) Tables;
4) Graphics;
5) Sound;
6) Speech;
7) Control signals for equipment.

Particular kind of output data is data for explanation of solution.

Separated task or subtasks may be performed by follow taxonomy:

- Recognition or classification. In this task the result is usually symbolic data, and this task usually can not viewed as sequence of similar subtasks, but may be viewed as a tree of classification subtasks;
- Acquisition of knowledge. This task basically is similar to recognition, but one may be composed from subtasks of different kinds;
- Reasoning or decision making. The result of solving of this task is symbolic data and this task can viewed as consecutive solving of different subtasks or tree of different subtasks;
- Calculation. In this task the result is value of any known function;
- Approximation. In this task the result is value of any unknown function;
- Prediction In this task the result is value of any unknown function from time;
- Planning. In this task the result is sequence of symbolic or numeric data in time;
- Explanation.

Every of above tasks has special performing and solving methods. And to select one of them we need respect to kind of input-output data and requirements to implementation (e.g., limits of resources, requirements to interfaces and so on). Then we must select kind of hybrid architecture with respect to system's requirements.

This description of taxonomy of different components for HIS development obviously is not full. One is oriented on local system, e.g. for development of intelligent system for any smart object (e.g., smart bed, intelligent agent based on mobile phone, mobile robot and so on). But for development of smart environment most important issue is to take account of distribution of smart objects and necessity to communicate among them. Therefore we need to introduce into methodology also parameters of communications, such as, parameters of signals, traffic, features of protocols. Besides, it is essential to understand and to describe somehow the relationship between distributed tasks (and its implementation) and features of communication. For example, is it possible and how to communicate between two neural networks in ubiquitous computing system? This problem is close to communication between human beings and evolution of natural language. These problems overstep the limits of this paper and show that this "introduction" is just start in a long way.

## CONCLUSION

Usage of hybrid approach to development of ambient intelligence in ubiquitous computing systems is obviously perspective way. This approach is bio-inspired and final goal of researchers is development of human-like mind implementing advantages of artificial intelligent system (may be phantom), such as, high speed and capacity of memory, easiness of communication with equipment, without loss of flexible and learnable of natural mind.

Future of ubiquitous computing with ambient intelligence may be viewed as appearance of thinking and speaking home or office (or shop) building consisting of thinking and speaking (may be not all) smart objects including assistive robots. Thus human person will deal with artificial characters being able help him to access different capabilities of building to improve comfort of life and efficacy of job. May be two ways to implement such ambient intelligence: to make separated intelligent agents (artificial persons) collaborating by communication or to centralize artificial person. In last case all smart objects may be viewed as parts of this person for perception, communication and action.

## REFERENCES


Baldauf, M., Dustdar, S. & Rosenberg F. (2007). A Survey on Context-Aware Systems. *Int. J. Ad Hoc and Ubiquitous Computing*, 2(4), 263-277.

Beetz M., Kirsch A., & Muller A. (2004). $RPL_{LEARN}$: Extending an Autonomous Robot Control Language to Perform Experience-based Learning. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS 2004* (pp. 1022 – 1029). New-York: ACM.

Bishop C. (1995). *Neural Networks for pattern recognition*. Oxford, UK: Clarendon Press.

Chen G. & Kotz D. (2000). *A survey of context-aware mobile computing research* (Tech. Rep. TR2000-381), Dept. of Computer Science, Dartmouth College.

Funabashi M. Maeda A., Morooka Y., Mori K. (1995). Fuzzy and Neural Hybrid Expert Systems: Synergetic AI. *IEEE Expert*, 10(4), 32–40.

Gavrilov A.V. (1988). Dialog system for preparing of programs for robot, *Automatyka*, v.99, Glivice, Poland, 173-180 (in Russian).

Gavrilov A.V. (2003). A combination of Neural and Semantic Networks in Natural Language Processing. In Hong-Hee Lee (Ed.) *Proceedings of the 7th Korea-Russia International Symposium KORUS-2003: Vol. 2* (pp. 143-147). Republic of Korea: University of Ulsan.

Gavrilov A.V. (2003) The principles of action of intelligent systems. In *Proceedings of International Conference on Information Systems and Technologies IST-2003: Vol.3*. (pp. 91-94), Novosibirsk, Novosibirsk State Technical University.

Gavrilov A.V. (2007) The principles of action of intelligent systems. In G. Marchetti (Ed.) *Mind, Consciousness and Language*. Retrieved from http://www.mind-consciousness-language.com/articles.htm.

Gavrilov A.V. (2008). Context and Learning based Approach to Programming of Intelligent Equipment. In Jeng-Shuang Pan, P. Kellenberger (Eds.) *The 8th International Conference on Intelligent Systems Design and Applications ISDA-2008* (pp. 578-582). IEEE Computer Society.

Gavrilov A.V. (2008). Hybrid Rule and Neural Network based Framework for Ubiquitous Computing. In Jinhwa Kim, Dursun Delen, Park, Franz Ko, Yun Ji Na (Eds.) *The 4th International Conference on Networked Computing and Advanced Information Management: Vol. 2* (pp. 488-492). IEEE Computer Society.

Gavrilov A.V. (2008). Usage of Neural Networks in Ubiquitous Computing Systems. In N.V. Pustovoy (Ed.) *Proceedings. of the 3rd International Forum on Strategic Technologies IFOST-2008*. Novosibirsk: Novosibirsk State Technical University.

Gavrilov A.V. (2009). New Paradigm of Context based Programming-Learning of Intelligent Agent. In A. Pascoal, V. Ufranovsky (Eds.) *Proceedings of 1st Workshop on Networked embedded and control system technologies. In conjunction with 6th International Conference on Informatics in Control, Automation and Robotics ICINCO-2009 (*pp. 94-99). Portugal: INSTICC Press.

Gavrilov A.V., Gubarev V.V., Jo K.-H. & Lee H.-H. (2004). An architecture of hybrid control system of mobile robot. *Mechatronics, Automation, Control*, 8, 30-37.

Gavrilov A.V., Gubarev V.V., Jo K.-H. & Lee H.-H. (2004). Hybrid Neural-based Control System for Mobile Robot. In Y.P. Pokholkov (Ed.) *Proceedings of the 8th Korea-Russia International Symposium KORUS-2004*: Vol. 1 (pp. 31-35). Tomsk: Tomsk Polytechnic University.

Gavrilov A.V., Lee S.-Y. (2007). Usage of Hybrid Neural Network Model MLP-ART for Navigation of Mobile Robot. In de-Shuang Huang, Luonan Chen (Eds.) *International Conference on Intelligent Computing* ICIC-2007, LNAI 4682 (pp. 182-191). Berlin, Heiderberg: Springer-Verlag.

Gavrilov A.V. & Chistyakov N.A. (2005). An architecture of the toolkit for development of Hybrid Expert Systems. In Yu, I. Shokin, O.I. Potaturkin (Eds.) *Proceedings of* The Second IASTED International Multi-Conference Automation, Control and Information Technology *ACIT-2005*. Novosibirsk: ACTA Press.

Gavrilov A.V. & Novickaja J.V. (2001). The Toolkit for development of Hybrid Expert Systems. In Y.P. Pokholkov (Ed.) *Proceedings of the 5th Korea-Russia International Symposium KORUS-2001: Vol. 1*. (pp. 73-75). Tomsk: Tomsk Polytechnic University.

Goertzel B., Pennachin C. (Eds.) (2007). *Artificial General Intelligence*. Berlin, Heidelberg, Germany: Springer-Verlag.

Goonatilake S. & Khebbal S. (Eds.) (1995). *Intelligent Hybrid Systems.* San Francisco, CA: Wiley.

Hochreiter S., Schmidhuber J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735-1780.

Hudak P., Courtney A., Nilsson H. & Peterson J. (2002). Arrows, Robots, and Functional Reactive Programming. *LNCS*, 2638, 159-187.

Hung N.Q., Shehzad A., Kiani S.L., Riaz M., Ngoc Kim Anh, & Lee S.-L. (2004). Developing Context-Aware Ubiquitous Computing Systems with a Unified Middleware

Framework. In Laurence T. Yang (Ed.), *The 2004 International Conference on Embedded & Ubiquitous Computing* (pp. 672-681), Berlin, Heiderberg: Springer-Verlag.

Jacobsen, H.-A. (1998). A generic architecture for hybrid intelligent systems. In *Fuzzy Systems Proceedings, IEEE World Congress on Computational Intelligence; Vol. 1* (pp. 709-714).

Lauria S., Bugmann G., Kyriacou T, Bos J. & Klein E. (2001). Training Personal Robots Using Natural Language Instruction. *IEEE Intelligent Systems*, 16 (2001) 38-45.

Mahalik N. P. (Eds.). (2007). *Sensor Networks and Configuration*. Berlin, Heidelberg, Germany: Springer-Verlag,

McGarry K., Wermter S. & MacIntyre J. (1999). Hybrid Neural Systems: From Simple Coupling to Fully Integrated Neural Networks. *Neural Computing Surveys*, 2. 62-94.

Medsker L.R. & Bailey D. L. (1992). Models and Guidelines for Integrating Expert Systems and Neural Networks. In A. Kandel and G. Langholz (Eds.), *Hybrid Architectures for Intelligent Systems* (pp. 154-171). Boca Raton, USA: CRC Press.

Meynard J.-P. (2000). Control of industrial robots through high-level task programming, Thesis, Linkopings University, Sweden.

Murphy R. (2000). Introduction to AI Robotics. MIT Press.

Pembeci I. & Hager G. (2002). A comparative review of robot programming language. Technical report, CIRL Lab.

Rabunal J.R., Dorado J. (2006). *Artificial Neural Networks in Real Life Applications*. Hershey, PA: IDEA Group Publishing.

Saad Liaquat Kiani, Maria Riaz, Yonil Zhung, Sungyoung Lee, & Young-Koo Lee (2005). A Distributed Middleware Solution for Context Awareness in Ubiquitous Systems. In *Proceedings of 11th IEEE International Conference on Embedded and Real-time Computing Systems and Applications* (pp. 451-454). IEEE Computer Society.

Samaka M. (2005). Robot Task-Level Programming Language and Simulation. In *Proc. of World Academy of Science, Engineering and Technology*: Vol. 9.

Satyanarayanan, M. (2001). Pervasive Computing: Vision and Challenges, *IEEE Personal Communications*, August 2001, 10-17.

Schilit B., Adams N., & Want R. (1994). Context-aware computing applications. In *IEEE Workshop on Mobile Computing Systems and Applications*. Santa Cruz, CA: IEEE Computer Society.

Seabra Lopes L., Teixeira A., Rodrigues M., Gomes D., Teixeira C., Ferreira L., Soares P., Girão J. & Sénica N. (2003). Towards a Personal Robot with Language Interface. In *Proceedings of 8th European Conference on Speech Communication and Technology EUROSPEECH'2003* (pp. 2205—2208). Geneva, .

Spiliotopoulos D., Androutsopoulos I. & Spyropoulos C.D. (2001). Human-Robot Interaction based on Spoken Natural Language Dialogue. In *Proceedings of the European Workshop on Service and Humanoid Robots (ServiceRob '2001)* (pp. 123-128). Santorini, Greece.

Tarik-Ul Islam Kh., Sarkar Jehad, Hasan Kamrul, Huq M. Rezwanul, Gavrilov A. V., Lee Y.-K., & Lee S.-Y. (2008). A Framework of Smart Objects and their Collaboration in Smart Environment. In Hyeong Ho Lee (Ed.) *The 10th International Conference on Advanced Communication Technology: Vol. 1* (pp. 852-855), IEEE Computer Society.

Vajda F. & Urbancsek T. (2003). High-Level Object-Oriented Program Language for Mobile Microrobot Control. In *IEEE Proceedings of the conference INES 2003*, Assiut - Luxor, Egypt, IEEE Computer Society.

Wahl F. M., Thomas U. (2002). Robot Programming - From Simple Moves to Complex Robot Tasks. In *Proceedings of First Interntional Colloquium "Collaborative Research Center 562 – Robotic Systems for Modelling and Assembly"* (pp. 249-259), Braunschweig, Germany.

Wan Jun (Ed.) (2001). *Computational Intelligence in Manufacturing Handbook,* CRC Press LLC.
Weiser, M. (1991). The Computer for the 21st Century. *Scientific America,* Sept. 1991, 94-104.
Wermter S. & Ron Sun, (2000). *Hybrid Neural Systems*. Heidelberg, New-York, Germany: Springer.
Williams B.C., Ingham M.D., Chung S.H. & Elliott P.H. (2003). Model-based programming of intelligent embedded systems and robotic space explorers. In *Proceedings of IEEE: Special Issue on Modeling and Design of Embedded Software* (pp. 212-237). 91(1), IEEE Computer Society.
Zadeh L.A. (1975). The concept of a linguistic variable and its application to approximate reasoning. *Inform. Sci.*, 8, 43-80.

**ADDITINAL READING**

Basten T., Geilen M. & de Groot H. (Eds.) (2003). *Ambient intelligence: impact on embedded system design.* NewYork, Boston, Dordrecht, London, Moscow: Kluwe Academic Publishers.
Bunke H & Kandel A. (Eds.) (2002). *Hybrid methods in pattern recognition*. Singapore: World Scientific Publishing Company.
Gavrilov A.V. Hybrid intelligent systems. Novosibirsk, Russia: Novosibirsk State Technical University (in Russian).
Hoya Tetsuya (2005). *Artificial mind system.* Berlin, Heidelberg, Germany: Springer-Verlag.
Jang J.-S. R., Sun C.-T. & Mizutani E. (1996). *Neuro-fuzzy and soft computing*. Upper Saddle River, NJ: Prentice Hall.
Jones M. Tim (2008). *Artificial intelligence: a systems approach.* Hingham, MA: Infinity Science Press LLC
Konar Amit (2000). *Artificial intelligence and soft computing*. London, New-York: CRC Press.
Melin P. & Castillo O (2005). *Hybrid intelligent systems for pattern recognition using soft computing.* Berlin, Heidelberg, Germany: Springer-Verlag.
Minsky M. (2006). *The emotional machine*. New-York, NY: Simon & Shuster.
Munakata Toshiniri (2008). *Fundamentals of the new artificial intelligence*. London, UK: Springer-Verlag.
Negnevitsky M. (2002). Artificial intelligence: a guide to intelligent systems. Harlow, UK: Addison-Wesley.
Ovaska Seppo J. (Ed.) (2004). *Computationally Intelligent Hybrid Systems: The Fusion of Soft Computing and Hard Computing.* Wiley-IEEE Press.
Remagnino P., Foresti G.L. & Ellis T. (Eds.) (2005). *Ambient intelligence: a novel paradigm.* Boston, MA: Springer.
Weber W., Rabaey J.M., Aarts E. (Eds.) (2005) *Ambient intelligence.* Berlin, Heidelberg: Springer.
Zhang Z. & Zhang C. (2005). *Agent-based hybrid intelligent systems. LNAI,* 2938, Berlin, Heidelberg, Germany: Springer-Verlag.
Zomaya Albert Y (Ed.) (2006*). Handbook of nature-inspired and innovative computing.* Berlin, Heidelberg: Springer.