TIEMS98: Disaster and Emergency Management: International Challenges for the Next Decade. The Fifth Annual Conference of The International Emergency Management Society, Washington, D.C. , May 19-22, 1998

Integrated Parallel Bottom-up and Top-down Approach to the Development of Agent-based Intelligent DSSs for Emergency Management

Adam M. Gadomski, Claudio Balducelli,
Sandro Bologna and Giovanni DiCostanzo
ENEA, CR Casaccia, 00060 Rome, Italy
+39-6-3048-3404 (voice)
+39-6-3048-6511 (fax)
gadomski_a@casaccia.enea.it,

Keywords: emergency, management, decision-support, IDSS, design , bottom-up, top-down, multi-agent system, intelligent agent.

Abstract

The paper presents on-going ENEA's results in the development of active/intelligent DSSs (IDSS) for emergency managers. The emergency management problem is initially analyzed top-down, from the perspective of the general functional requirements of IDSSs, and integrated with the bottom-up perspective of incremental prototyping. Independently developed autonomous agents support higher, 'mental' functions of IDSS. Such parallel bottom-up and top-down development of a generic IDSS kernel, supported by intelligent multi-agent architecture, enables:

- various real-time specialization of the system on the level of tools,
- strong reduction of the design time by parallel execution of project phases
- easier the verification and validation of the system as independent tasks .

The advantages of this methodology are illustrated by the current ENEA's results related to the project GEO.

1. Introduction

The current paper is a continuation of the ENEA's studies related to the development of a multipurpose Intelligent Decision Support System (IDSS). The project is realized under the umbrella of the ENEA’s long term MINDES Program synchronized with other worldwide programs and accepted as an Italian contribution to the GEMINI (Global Emergency Management Information Network Initiative) of the G7 Committee [Gemini,96]. The main objective of MINDES is to develop a multipurpose intelligent Decision Support Systems for industrial emergency management. The technology involved is based on the novel intelligent agent approach which should not only provide data (selected according to situation assessment and intervention procedures from emergency plans) but to provide an active decision support related to the choices of adequate actions. A definition of the MINDES Program is included in the GEMINI’97 Proceedings [Bologna,Gadomski,96]. Local decision support systems should be, in future, connected with another similar emergency management centers in frame of an international Global Emergency Management Information Network. The MINDES program scope is to reduce the probability of human managerial errors during high - risk decisions by the development of advanced decision-support and operator training systems.
An initial conceptual framework for such systems has been theoretically defined in the paper [Gadomski at al.,95].

The next steps, supported, in parallel, by development of prototypes, have been performed in three strategic directions:

1. ***Bottom-up identification and design*** of specific functions required by the emergency managers with predefined roles, such as oil port director, supervisor of the node of an oil pipes network.
2. ***Top-down modeling*** of a generic frame scenario of emergency manager activities; role independent
3. ***Development of intelligent agents*** which may support mental functions of decision-makers in an autonomous manner.

After our experiences in projects ISEM[Sepielli at al.,89], MUSTER [Balducelli at al,95], CIPRODS [Gadomski,Di Costanzo,96], GEO [Balducelli at al,97], we discovered that the first difficulty in the design of IDSSs for emergency managers, are not referred to software technologies but to the vagueness and incompleteness of end-user requirements. The potential users of IDSSs are practitioners and they have serious problems with mental structuring and description of their own activity in the form of a complete set of abstract categories covering the representation of an emergency domain, as well as, their various temporal and permanent constrains.
Up to now, for the reason of the problem complexity and possibility of appearing unexpected decisional situations, this problem is also not resolved yet by knowledge engineers and system modelers.
The complexity of emergency-operator's decisions is characterized by the necessity of interactions with different domains in order to choice one action adequate to the current situation, and to execute it in the proper way. A final intervention act should mitigate emergency situation in a preselected real world emergency domain, such as populated territorial zone, industrial complex, port or railway station.
Therefore, in the context of IDSS, we think only about an action-oriented decision-making [Gad.90]. Such type of decisions consists of the preparation of the various data for a choice of an intervention and on its execution by the activation of available tools. In general, one emergency manager decision requires various activities related to different objects from the manager environment. The decisional context, its numerous domains are illustrated on the Fig. 1.

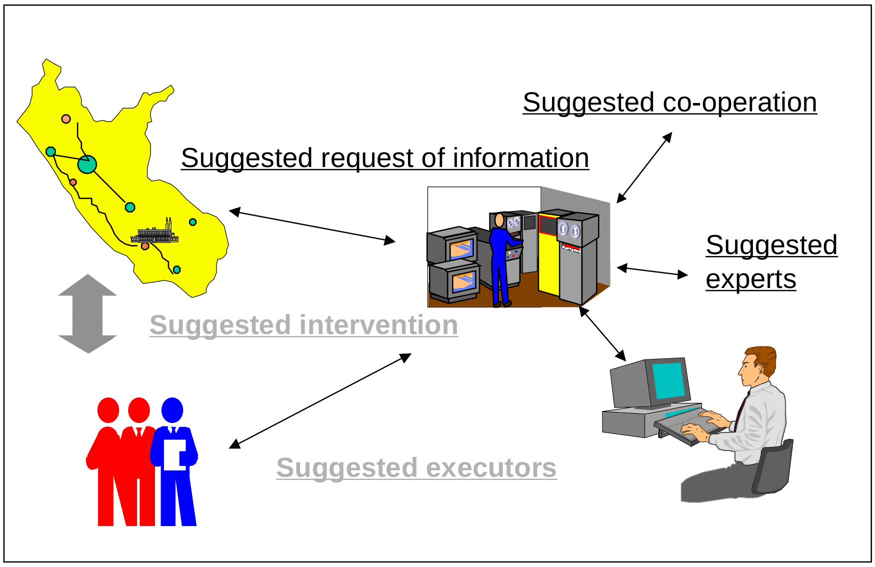


Fig.1 Possible domains of intervention of IDSSs [Gadomski.97].

From different models of more or less generic emergency management scenarios investigated in ENEA a confirmation of the basic repetitive cycle of ***decision-oriented activities*** results. It is divided on the following ***generic phases*** [Gadomski at al., 95] :

P1. current situation assessment,
P2. cause searching,
P3. consequences prediction/assessment,
P4. intervention planning,
P5. current action execution.

In concrete situations, some of them are omitted or realized in a different range by their decomposition into conscious processes of decision-makers, for example, in well structured emergency domain, as in an engineering plant, the above phases, in practice, can be supported by the following cycle of the computer activities (executed by activated functions):

- detection/recognition of the abnormal event - it is an initial activity of the **situation assessment**
- choice of adequate procedures - it is a **task planning** but reduced to a choice from an available plans (procedures) base,
- choice of an intervention action - it is an **action planning** but only reduced to a choice from known alternative actions under predefined fixed criteria.
- demonstration to the user a set of direct intervention tools necessary for this action execution, as telephone numbers, location of resources, etc.

Here, the activities above were recognized as difficult but **possible** to the formalization as "an ideal operator" *tasks*, and were **allocated** to the IDSS as its main *production functions*. The remaining activities from the generic scenario were allocated intuitively to the system end-user.
In general, IDSS functions represent those activities which are possible to formalize and, in parallel, are considered not trivial for execution by human emergency managers in a concrete class of emergency situations. In such context, the design of IDSS can be performed top-down or bottom-up.

2. Bottom-up Design of IDSS

Bottom-up design is an incremental approach applicable for the development of qualitatively new systems where their application range and complexity of functions can not be defined on the base of their future user requirements. This approach has strongly explorative character and it is relies on the verification of the utility and applicability of new software methods and technologies for never yet implemented particular functions.

In the case of active DSSs, the bottom-up approach relies on:

1. ***recognition of modelable activities*** in a concrete decision-making activity of the manager/operator. The criteria adopted: activities and their results must be well visible by an external observer, usually related to separable types of events or available intervention procedures and used resources (such as access to databases).
2. ***their implementation*** as functions of a DSS.

The DSS system, in this way constructed, **must be menu-driven**, because any sufficiently detail generic scenario of the user behavior don't exist formally and the user choice of a designed functions depends on his/her mental processes and informal professional experience. The activation of particular functions are usually driven by external foreseen events according to the scheme***:***

detected symptom or information about known type of events -> <u>necessary</u> preplanned reaction,

where the necessity is recognized by the human user.
The symptom could be, for instance, fire detection, information about explosion of an oil container or about intoxication of the population in a certain territory.
Here, the modeler effort is mainly concentrated on the taking under consideration a maximal number of possible details related to the emergency event, and on the application of maximally sophisticate reactions

using available advanced software technologies, such as numerical simulations, neural-fuzzy or genetic algorithms.

Unfortunately, independently on how perfectly are executed separate support functions, in the real emergency management, the user always can be in situations unexpected by the system designer.These systems have no possibility of learning because they are to precise for self-modifications.
Their paradox relies on the specialization: better function $\rightarrow$ less modifiable function.
As the consequence, to reduce domain of application of the system and, subsequently, to design new systems for every next emergency type, emergency organization, kind of territory and a new user, are the designers tendency. The following designer strategies are here involved:

1. Increase the detail of modelable interventions
2. Increase the number of modeled functions
3. Perfectioning of the intervention reactions.

Using the event model we receive a design tree having continuous tendency to unforeseen increase in all directions, the Fig.2. In this situation it is very difficult to foreseen the final state of the designed system in frame of the project time schedule constrains.

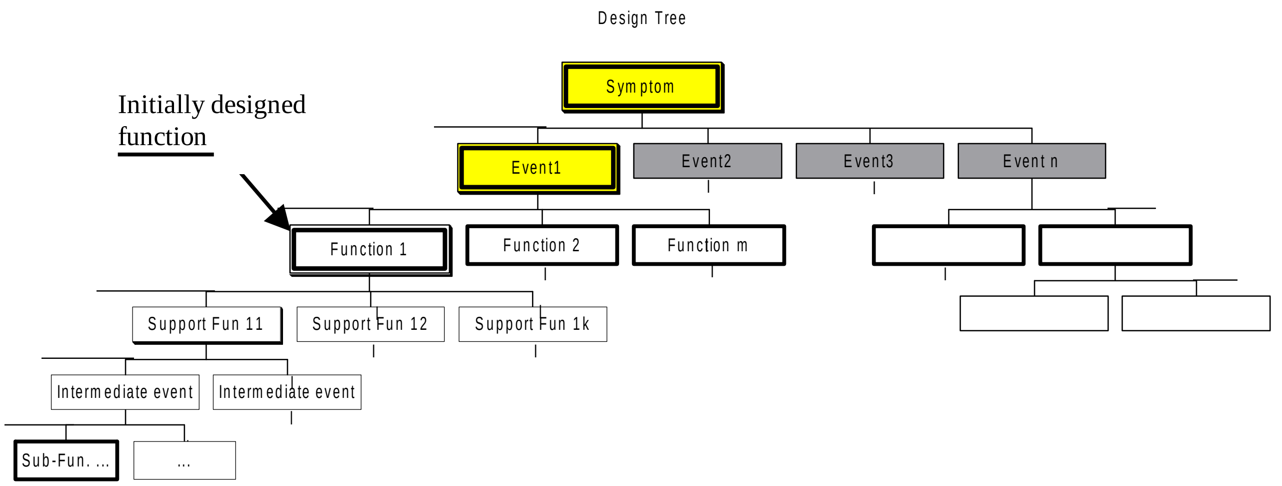


Fig. 2. Bottom-up development of active DSS.

Such problem is invariant ever the functions are grouped, by the designer, according to a top functional architecture and ever they are based on object-oriented clustering; according to the rule that one object may be involved as a tool for many functions/subfunctions.
Summarizing, to improve the bottom-up design, some top-down constructed constrains are required. The range and strategy of top-down approach lead to the design of more or less active/intelligent DSS.
In our work, bottom-up design has been focused on the following aspects:

- ***perfectioning*** of the preselected tools and
- ***facilitation*** of the development of new functional tools, in frame of integrated Emergency Management Active Tools-Kit module (EMAT**).**

3. The Emergency Management Active Tool-Kit in GEO system

The GEO interface [Balducelli at al..97] includes the Emergency Management Active Tools-Kit module (EMAT**).** It is a set of operational/procedural knowledge involved in the real-time management and control of severe accidents inside petrol-chemical plants.
The interface was developed using a bottom-up incremental implementation strategy: it means, starting from a well defined and structured emergency domain and emergency manager role.

The EMAT uses also a set of numerical and discrete simulation models of the most important events in a considered physical domain.
The software architecture was generalized to produce an EMAT adaptable to a wide class of interventions.
The interface has been designed and developed using visual programming techniques and making use of the functional approach to display and manage information.
The same type of tools-set could also be used in plant operator's training systems especially to improve the utilization of the emergency resources during the main forecast emergency events.

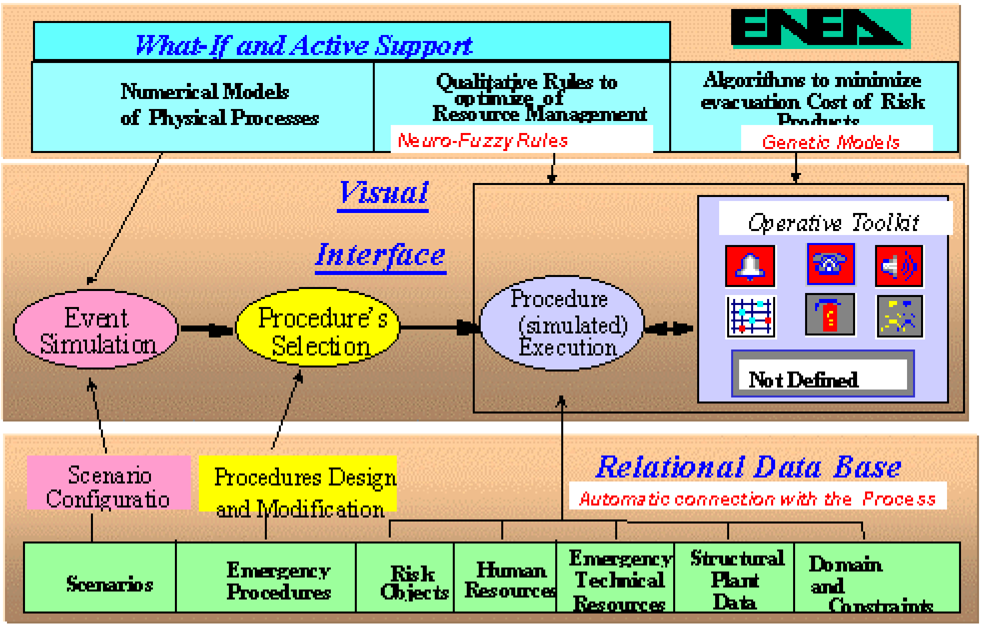


Fig. 3 EMAT functional interrelations inside the GEO system

The Emergency Management Active Tool-Kit (EMAT) is structured as a shell connected with a Relational Data Base, on one side and with the outputs coming from numerical models of the physical domain on the other side. It has been used, as visualized in the Fig. 3 inside the GEO system prototype.

Configuration of Emergency Scenarios

GEO allows the user to configure and run Emergency Scenarios supporting the set-up of *primary* and *secondary* data of the physical domain:

- *primary data* are data that directly characterize the emergency scenario as the type of event, the data and time of the event, the involved plant systems and objects etc.
- *secondary data* characterize the external conditions in which the event occurs as meteorological conditions and plant operative statuses.

Emergency Procedure's Design

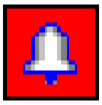
GEO contains an appropriate *Procedure's Editor* that, during system set-up sessions, allows the user to design and implement emergency procedures, and to connect procedures to the different types of configured emergency events.
The procedure's structure is subdivided into *phases* (plan steps associated to the realization of more general operative goals) and every phase can be divided into several *tasks* (plan steps associated to the realization of more elementary operative goals).
Explanation texts and *precedence* constraints can also be associated to the procedure's tasks and phases.
The Procedure Editor allows the user to *link* events (primary data in the configured scenarios) to procedures.
More types of events can be linked to the same procedure.

EMAT Toolkit Menu Bar

The right execution of a single procedure task can be done in several way [Azvine at al.,91], [Yaverbraum,94] by different operator *actions* depending about the external domain statuses and conditions. Every action execution is supported using an adequate *operative tool*. Tools become active and can be used by the operator by selection from the graphical *Operative Toolkit Menu Bar*. A single tool, using the above mentioned procedure's editor, may be associated to several tasks inside the same procedure realization.

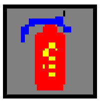
In an object oriented perspective, operative tools are like *methods* activated in frame of an action execution. The shell actually contains more general (not domain oriented) tools and more specific tools (domain oriented):

The following principal domain oriented tools (relative to an oil-chemical plant type) were defined:



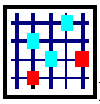
Support the operator to *activate* the general alarm

INSITE PLANT ALARM



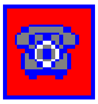
Support the operator to *optimise* the utilisation of the fire-proof resources

FIRE-PROOF SYSTEM



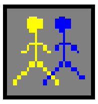
Support the operator to *minimise* the time needed to evacuate the risk products from the oil tanks

MANIFOLDS



Support the operator to *find* the telephone numbers of the 'Outside Authorities to be Alerted'

OUTSIDE PLANT ALARM



Support the operator to *find* the on-duty emergency personnel

PERSONNEL MANAGEMENT

These tools can be easily inserted and removed from the *Operative Toolkit Menu Bar*.

What-if and Active Support Functions

As it is visualized in the Fig. 3, the tools are supported by a set of What-If active support functions. In particular, in the generic emergency domain, the following three functions are defined:

1) Numerical Process Models of the Physical Domain (to predict thermal radiation levels and temperature behaviors on risk objects)
2) Qualitative Rules to optimize the Emergency Resource Management (to optimize the utilization of fire-proof systems;
3) Algorithms to minimize the evacuation Cost of the Risk Products (to reduce the time, the risks, the economic losses during the evacuations of products and items that are a risk condition).

As it is visualized in the Fig. 3, for functions 2 and 3, a white box contains the identifications of AI methodologies that will be utilized to make more active and efficient the function itself:

- Neuro-Fuzzy Rules Generation: it means *to improve* a set rules, acquired from Experts of Domain Knowledge, through an *off-line analysis* of the process simulators results: the neural networks will be used to learn and to extract the more predominant behaviors from a set of pre-defined typical configurations of the event and the risk involved objects; the fuzzy algorithms will be used to discrete the obtained thermal behaviors and to produce fuzzy rules to be used on-line inside the system.
- Genetic Models: a genetic type of processing will be used to produce the more efficient solutions to evacuate the risk products far away from the event location. They must minimize the evacuation time taking in account a set of constraints as the availability of the target places and the minimization of the economic damage.

The Fig 4 shows the logic interrelations between the GEO models involved in the off line and on line sessions.

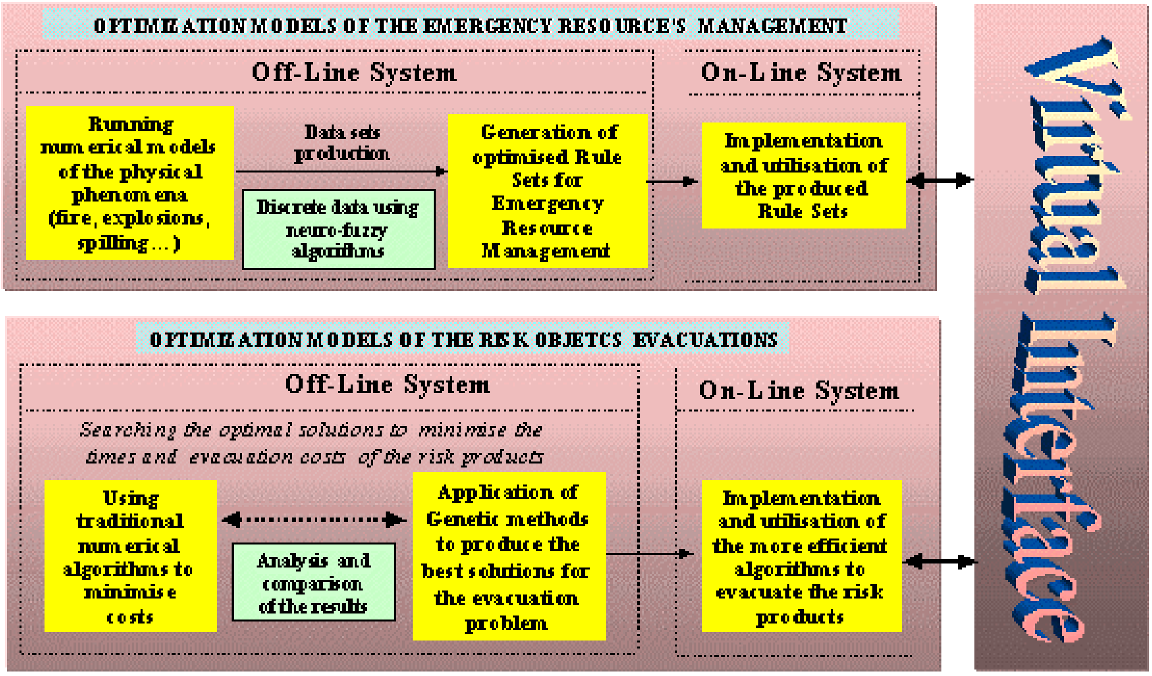


Fig. 4 Application of neuro-fuzzy and genetic algorithms inside GEO system [Balducelli at al.,97]

A Virtual-Reality Tool

As it was described above, the procedural system allows the operator to execute operative actions by the support of an adequate *operative tool*. Moving and managing items and resources inside very hostile surroundings is not a task to be easily simulated and learned.
A virtual reality system should be in the future introduced as a new tool inside the Tool-Kit. Using this tool the operator can execute simulated immersions inside the three-dimensional environment discovering various possible ways to use, for instance, the fireproof mobile resources.
The conceptual design of visual *what-if* simulation and its integration with the previously realized decision support functions will be a difficult research task.
In general, a construction of advanced IDSSs requires a top-down approach to its conceptual design.

4. Top-down Design and Agent-based IDSS

Contradictory to the bottom-up is a top-down design. The top-down design can be seen as follow:

- ***top-down identification*** of sub-activities by decomposition of the generic phases of decision-making (for the chosen class of emergency management) up to the level where selected sub-activities are considered formally modelable
- ***modeling and formalization*** of the activities
- ***definition of the necessary user-computer interface*** for their automatic execution; in frame of generic decision-making scenario.

The design goal of human-computer system is decomposed on sub-goals linked with functions, and functions are properties of designed calculation processes. The processes are realized by a software system, i.e. in frame of the system structure (architecture). The TOGA conceptual framework of this type of top-down decomposition is presented in the papers [Gadomski,94], [Gadomski at al.,95]. Here, it is necessary to recall some key definitions and explanations.

Design goal represents a state of the designed-system world which designer expects to achieve.

Functions are such properties of the system, which are necessary for the design-goal achievement.

Calculational processes are processes executing by calculational algorithms and different uniquely described computational methods.

The same function is possible to realize using different calculational processes, for example, using different software technologies.

Architecture (system-structure) is a framework of invariant relations among system components.

Intervention-goal is an expected state of the domain, which activate action-oriented decision-making processes of the system, in the specific situation of the domain.

Task is an action property necessary/requested for intervention-goal achieving. *Action* is a situation dependent realization of a task.

Therefore, in different domain states, the same task can be executed by various actions.

In the GEO system, the emergency management generic design goal is represented as N alternative design goals. Every goal is defined by a class of the emergency domain states (dependent on specific initial conditions).
The N goals achieving requires N intervention realization functions. Every function is realized by predefined procedure. These procedures require a sequence of ordinate functions being executed by the IDSS and the operator together.

Application of the GEO System to Support Emergency Operator

As it is visualized on the Fig. 3, the GEO system realizes three main separable functions:

Configuration of Emergency Scenarios
On-line support in the realization of intervention procedures and
S*imulation of emergency scenario*.

The *on-line support function* of GEO is decomposed and allocated in frame of a general, top-down defined functionality of the GEO-operator aggregate, what is illustrated in on the Fig.5.
The following scenario presents that allocations.

- F1. ***System*:** choice of the adequate procedure, from a structured Procedure Bases, on the base of initial information about a current symptom.
- F2. ***System*:** demonstration of the procedure phases included tasks, where the order of the tasks execution depends on the user choice.
- F3. ***User*:** <u>choice of a task</u>
- F4. ***System*:** demonstration of icons of available tools
- F5. ***User*:** <u>choice of an action</u>, because the tasks can be performed by different actions.
  The choice of one action is not modeled mental process. It must be concluded by the activation one or more of the available tools.
  F6. ***System:*** action execution by production and realization of commands that modifies domain event attributes.

<u>Remarks</u>: Here, the complexity of tools is not important; using bottom-up approach, every tool is developed according to the detailed knowledge about one type of intervention, and according to the calculational possibilities of the employed methods, for instance, they can be a genetic or neural-fuzzy algorithm too.

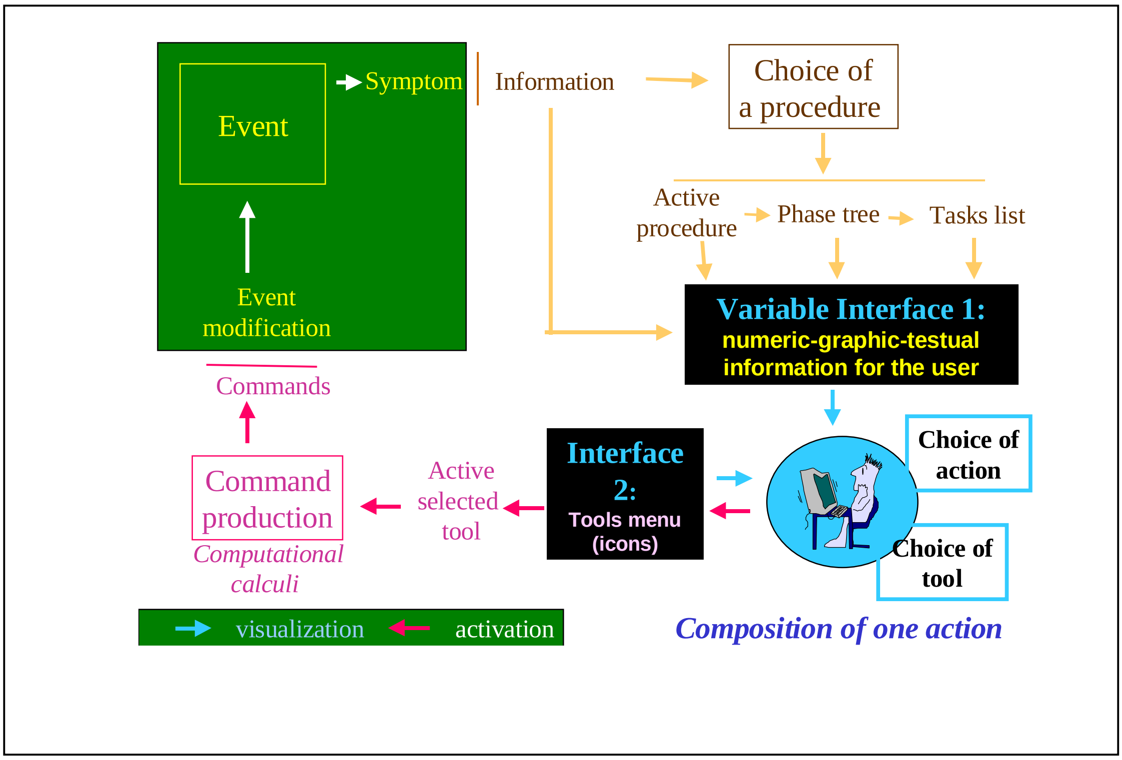


Fig. 5 Schema of functions allocation among active DSS and its user [Balducelli,Gadomski,97]**.**

It was verified that EMAT can be configured/integrated with multi-agent architecture of IDSS, improving the autonomy and flexibility of the computer - human communication protocols.
In frame of such framework, the functions can be also supported by autonomous agents. Adequately to their complexity of their tasks, the agents can be more or less "intelligent".

Intelligent Agent in IDSS Design

Application of the intelligent agent paradigm to the design IDSS means, to apply human metaphor [Wooldridge,Jennings,95] i.e. to see IDSS as a "worker" equipped with various tools adequate to his own role. He can interact with physical systems but mainly; "he" communicates with various active and passive information sources in order to realize tasks.
Therefore, intelligent agent activity relies on:

- recognition of its own task
- choice/plan an adequate actions
- realization of this tasks using available tools.

The general IPK (Information, Preference, Knowledge] architecture and functionality of an abstract intelligent agent is described in [Gadomski,93], [Balducelli,Gadomski,93], [Gadomski, Zytkow,94], [Gadomski at al.,95], [DiCostanzo, Gadomski,97] but other agent architectures can also be taken under consideration, see for example [Singh,94], [O'Hare,Jennings,95], [Sycara,96] or [Dunin,Treur,95], the last is related to BDI (Belief, Desire, Intention) agents. However, independently on a concrete agent structures assumed, the following generic functional components of an intelligent agent is useful to distinguish:

1. Communication (external/interface) layer
   1.1 preceptors zone - c. with non agent equipments
   1.2 effectors zone - c. with non agent equipments
   1.3 messages zone - c. with other agents

2. Domain-reasoning layer
   2.1 associations stratum (skill reasoning - neural-fuzzy techniques)
   2.2 comprehension center (allocation of information to agent ontology i.e. to a model-based knowledge)
   2.3 preferences center; goal choice
   2.4 knowledge (model and operational knowledge) center; intervention/action choice
3. Meta-reasoning kernel/layers; it can include centers for strategies of: goals management, planning, learning, tutoring, etc.

Such architecture tries to be independent on agent types. According to the above structuring, agents without the kernel are not intelligent yet.
According to the IPK repetitive architecture, an intelligent agent (called personoid) is composed with simple agents [Gadomski,98]. Every agent has its own reasoning mechanism, role dependent access to information (communication protocols), knowledge and preferences on different meta-reasoning levels.
An advantage of expected personoids application relies on the possibility to exchange their domain knowledge and preferences according to the needs of the IDSS user.
Now, dependently on complexity of the system, the system functions can be allocated to one or more intelligent agents. The Fig. 6 illustrate situation when an action definition and tool choice, it means an action planning, are allocated to one intelligent agent. The planning is performed according to predefined domain and role-dependent constrains. In decisional situations, a concrete tool from the Operative Tools-Kit menu is suggested to the operator, by a software agent.
The conclusive operator decision relies on application of not formalized human criteria in the final choice. Of course, the operator must have an access to the source information by the Information interface and Operative Tool Menu Bar.

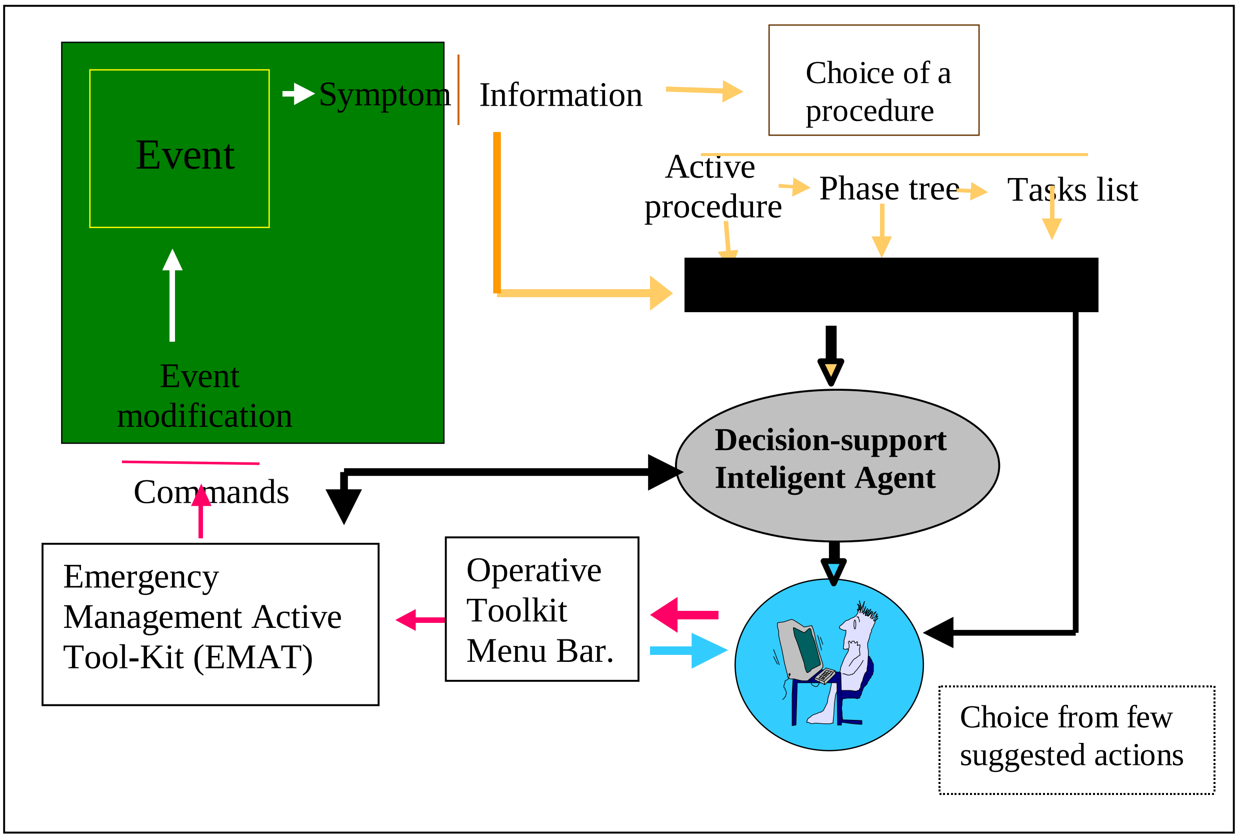


Fig. 6 An example of IDSS with one decision-support intelligent agent.

5. Conclusions: About Project Realization

An application of the bottom-up and top-down methodology supported by multiagent technologies, enables to copy with the next serious problem of the innovative development of IDSS for emergency managers. This is the project planning and management,

An IDSS is a Knowledge-Based type system, for this reason, it requires a specific *life cycle* differ from typical object oriented software systems. In this matter numerous literature sources are available.

However, independently on the software technology accepted, a realization of an innovative IDSS requires the following basic phases:

1. Preliminary recognition of possible end-users requirements and constrains
2. Choice of the emergency domain (with class of events) and roles of the system end-users
3. Knowledge acquisition (1) for/and modeling of a selected class of emergency management situations.
4. Allocation of modeled functions to the human users and to the planned IDSS.
5. Knowledge acquisition (2) for/and specification of a generic user-system communication scenarios and protocols.
6. Allocation of the functional models to the system components
7. Development of intelligent agents
8. Development of agents' and user's tools
9. System integration
10. System testing and validation using simple emergency test cases.
11. Modification of the system
12. User training.

It is necessary to stress that IDSSs are strongly heterogeneous, therefore the project must be supported by:

- project development software platform; agent-oriented languages [FIPA,98], [O'Hare,Jennings,96]
- project-oriented vocabulary (continuously updated)
- emergency management ontology and developers meta-ontology for a clear for the designer team, common conceptual -design methodological platform, such as TOGA, [UML] or [National Rose].

We consider the above meta-conceptual tools indispensable for the realization of the project.

In any case, the assumption of the top-down, bottom-up and agent-based perspectives should facilitate and strongly reduce the design time of IDSSs' prototyping. Such approach makes transparent system functional and structural architecture, as well as, enables, in parallel: top-down and bottom-up identification of the system functional scenarios, tool building and intelligent agent design.

References

[Azvine at al.,91] B. Azvine, N. Azarmi, K.C. Tsui. An Introduction to Soft Computing – A Tool for Building Intelligent Systems. Lecture Notes in Artificial Intelligence 1991, "Software Agents and Soft Computing", Nwana- Azarmi Eds. pp. 191-210.

[Balducelli,Gadomski,93] C.Balducelli, A.M. Gadomski. Intelligent Agents in Computer Supported Training for Emergency Management. In the Proceeding of the First International Round-Table on Abstract Intelligent

[Balducelli at al.95] C.Balducelli, S. Bologna,G. Di Costanzo,, A.M. Gadomski,G. Vicoli. Computer aided training for cooperating emergency managers: same results of MUSTER Project. Proceedings of the Membrain '95 Conference,Olso, 18-21 Giugno 1995.

[Balducelli at al.97] C.Balducelli, G.Vicoli, N.Arcidiacono. "EMAT (Emergency Management Active Tool-Kit) An Active Decision Support and Training System based on the Configuration and Simulated Execution of Emergency Scenarios", Halden Project Workshop Meeting on "Intelligent Decision Support Systems for Emergency Management" Halden, 20th-21st October, 1997

[Balducelli,Gadomski, 97] C. Balducelli, A.M.Gadomski, " Active Decision Support Interfaces for Emergency Management: Bottom-up and Top-Down Approaches". AIIA's Workshop on Intelligent Interfaces, Rome, Nov. 1997.

[Bologna,Gadomski,96] S. Bologna, A.M.Gadomski. Managerial Intelligent Node of Decision Support for Emergency Supervisory (MINDES) Program. The proceedings of the GEMINI Annual International Meeting Roma, 3-6 settembre 1996.
[Costanzo,Gadomski,97] G.D.Costanzo, A.M.Gadomski, A Prototype of an Active Decision Support System, based on an Abstract Intelligent Agent Architecture, TIEMEC 1997: Proceedings of "The International Emergency Management and Engineering Conference",1997.
[Dunin,95] B.Dunin-Klepicz, J.Treur "Compositional Formal Specification of Multi-Agent Systems", In Intelligent Agents, M.J.Wooldridge, N.R.Jennings (Eds.), Springer Verlag, 1995.
[FIPA] Agent Communication Language. The FIPA document. http://drogo.cselt.stet.it/fipa/spec/fipa7412.htm
[Gadomski,93] A.M. Gadomski, TOGA: A Methodological and Conceptual Pattern for Modeling Abstract Intelligent Agent. Proceedings of the First International Round-Table on Abstract Intelligent Agent (AIA '93). Rome., Jan. 23-27, 1993. , A.M. Gadomski (editor). Printed by ENEA, Feb.1994. 25-27 Gen., Rome, 1993, Published by ENEA, Feb.1994.
[Gadomski, Zytkow,94] A.M.Gadomski, J.M.Zytkow. Intelligent Agents: Paradigms, Foundations and Expectations. The Proceedings of the Second International Round-Table on Abstract Intelligent Agent, AIA94, Printed by ENEA, Rome, 1995.
[Gadomski at al.,95] A.M.Gadomski , S. Bologna, G. Di Costanzo. Intelligent Decision Support for Cooperating Emergency Managers: the TOGA based Conceptualization Framework. The Proceedings of "TIEMEC 1995: The International Emergency Management and Engineering Conference", J.D. Sullivan, J.L. Wybo, L. Buisson (Eds), Nice, May, 1995.
[Gadomski,Di Costanzo,96] A.M.Gadomski, G. Di Costanzo. Intelligent Decision Support System for Industrial Accident Management. The Proceedings of 8th European Simulation Symposium , SCS, Genoa, Oct.,1996.
[Gadomski.97] A.M.Gadomski. The Nature of Intelligent Decision Support Systems. Intelligent Decision Support Systems for EmergencyManagement. The HALDEN Project Workshop, Oct.20-21,1997, Transparent Sheets: http://erg2217.casaccia.enea.it/activities/emergency/eme-idss/sld001.htm
[Gadomski,98] A.M.Gadomski: Agent and Intelligence, http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-agen.html
[Gemini,96] http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-gemi.htm http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-prog.html
[O'Hare,Jennings,96] G.M.P. O'Hare, N.R.Jennings. Foundation of Distributed Artificial Intelligence, J.Wiley&Sons, 1996.
[Rational Rose] http://www.rational.com/demos/rose4demo/wlkthr_rse.html
[Singh,94] M.P.Singh, Multiagent Systems : A Theoretical Framework for Intentions, Know-How, and Communications. Springer-Verlag, 1994.
[Sepielli at al.,89] M.Sepielli, A.M.Gadomski,G.Brena, Information-Flow Model for Emergency Management System (Italian contrib. to the Final Report of Task 1.1), ENEA ,TERM-MEP, Internal Report, C.R.E. Casaccia, May,1989.
[Sycara,96] K.Sycara at al, Distributed Intelligent Agents In IEEE Expert: Intelligent Systems&Their Applications. Vol.11, N.6, Dec 1996.
[Yaverbraum,94] G.J. Yaverbaum, M.A. Reynolds. Managerial Problem Identifier (MPI): Integrating Expert System and Decision Support Technologies. Expert Systems With Application, Vol. 3, pp. 507-515, 1991.
[UML] Whitepaper: Unified Modeling Language for Real-Time Systems Design", http://www.rational.com/uml/hot and http://www.rational.com/uml/1.1.
[Wooldridge,Jennings,95] M.J.Wooldridge,N.R.Jennings (Eds.), Intelligent Agents, Springer Verlag,1995.