Towards User-centred and Cost-effective Development Of Environmental Decision Support Systems

Ion Leon Batachia

National Institute for R&D in Informatics
Decision Support Systems Research Laboratory
8-10 Averescu Avenue,
71316 Bucharest
ROMANIA
E-mail:leon@risc.ici.ro

Abstract: Environmental problems are complex and multi-disciplinary in nature. While the necessity of EDSSs - dedicated to support decision-making in environmental planning and management, is largely argued, it is also recognized that EDSSs, in general, are complex applications possible integrating different advanced technologies and requiring high research and development efforts. In the paper a framework for developing EDSS applications based on the results of the TRACE project (a research and development project funded by European Union) is briefly described. The framework represents a cost-effective alternative in developing EDSS applications also providing tools and guidelines for user-centred design. The TRACE approach in developing a specific EDSS supporting the user in post-windthrow management in Romanian forests is illustrated in the second part of the paper.

Keywords: Decision Support Systems, Environmental Decision Support Systems, User-Centred Design, Task Analysis, Constraint Satisfaction Problems, Resource Allocation, Post-Windthrow Management

1. Introduction

All managerial activities revolve around decision-making. The manager is first and foremost a decision -maker. Organizations, different in nature as they might be, abound with decision -makers at various levels.

The time frame of decision-making is shrinking while its global nature is expanding, necessitating the use of computerised support. One of the major technologies that provides computerised support for decision-makers is represented by decision support systems (DSS).

A DSS should support decision -making processes for unstructured or semi-structured tasks, i.e. in those contexts in which support that can be provided by traditional techniques and information technologies in use in organizational contexts like management information systems (MIS), electronic data processing (EDP) or operations research and management science is rather weak. The terms "unstructured" and "semi-structured" are used to characterize decision -making in case of complex or unspecified problems, for instance when the problem spans on long time periods, or the problem is related with uncertain factors or is characterized by a lack of knowledge or quantitative data. Several excellent books dealing with the DSS paradigm [Turban and Aronson, 1998], (Sprague and Watson, 1996], [Dhar and Stein, 1997] should be reminded.

A set of ideal characteristics of DSS is given in Turban and Aronson (1998): (i) deal with semi-structured decisions; (ii) are intended for various management levels of an organization; (iii) are adapted to both group and individual needs; (iv) support both interdependent and sequential decisions; (v) support all three phases of the Simon (1977) problem -solving process: intelligence, design and choice; (vi) support a variety of decision styles and processes which could occur among a group of managers; (vii) are adaptable and flexible to changes in both the organization and its environment; (viii) are easy to use and modify; (ix) improve the effectiveness of decision -making and problem solving, not only the efficiency; (x) allow humans to control the computer support and do not try to control the human problem -solving and decision-making; (xi) support an evolutionary usage and are easily adaptable to changing requirements; (xii) are easy to construct if the logic of the context can be formulated; (xiii) support modelling, and allow the use of knowledge.

A particular class of such systems is represented by environmental decision support systems (EDSSs) [Guarisso et al, 1989] which refer to decision -making in environmental planning and management. Environmental problems are complex and multi-disciplinary in nature requiring integration of the quantitative science and engineering components with socio-political, legislative and economic aspects. As the protection of the environment and the principle of sustainable development are high priority issues in most countries in our days, EDSSs are gaining an ever higher consideration by the public administrations and the scientific community. In this context, finding possible solutions to develop efficient and effective EDSS at a low cost it is an appealing and at the same time challenging area of research.

One frequently asked question in the software development field is how to build interactive systems that are useful for, easily validated and appreciated by the people using them. The best guarantee for success seems to be an approach to design, called user-centred design. User-centred design is an iterative process of developing a system step-by-step, by multi-disciplinary teams, and with the involvement of potential end-users.

One possible way of developing EDSS applications at a low cost in a user-centred approach is by fitting into the framework provided as a result of TRACE research project (an INCO-COPERNICUS project funded by the European Union within the ESPRIT Programme).

In the paper, the TRACE methodological framework for developing EDSS applications is briefly described, also illustrating the TRACE approach in developing a specific EDSS application.

2. Environmental Decision Support Systems

The objectives and characteristics of an EDSS follow from the more general ones of DSS and take into account some basic features, common to environmental problems. There are features related to natural phenomena (spatial and temporal features, periodicity, etc.) and also features related to the decisional process (multi-disciplinary information, distributed problem-solving responsibilities, multiple criteria).

Environment-related activities may be grouped into five general functional areas [Paggio et al, 1999]:

- prediction (risk analysis, forecasts);
- medium/long-term planning;
- monitoring and surveillance;
- crisis/emergency management;
- post-incident damage evaluation and reclamation.

While the necessity of EDSSs is largely argued, it is also recognized that EDSSs, in general, are complex applications possibly integrating different advanced technologies and requiring intense research and development efforts. The key to useful computer- based decision support is integration. As a basic concept integration recognizes , in any given software system for real-world applications, several sources of information or databases, more than one problem representation or model, and finally a multi-facetted and problem-oriented user interface ought to be combined in a common framework.

At the level of data and background information, numerous and often incompatible, non-commensurate information from disparate sources have to be brought together. Institutional, conceptual, and seemingly simple technical problems such as different units of measurement, different map projections, hard - to -trace paper files and missing documentation are some of the obstacles frequently encountered.

At the level of tools, there are several levels of integration, ranging from simple file transfer between different methods and programs to fully integrated systems. Geographical information systems, databases, complex simulation and optimisation models or artificial intelligence technologies are all typically potential candidates to the integration.

EDSSs are characterized by a complex approach involving several technologies, tools and devices, and dealing with different aspects of environment-related activities. A general perception is that to support the decision - making needs in solving complex environmental problems is to afford the development and use of appropriate EDSSs. This is a major obstacle in a wider development and usage of such systems in practice.

One of the objectives of the TRACE project was to provide a framework for designing and developing at a low cost of a wide range of effective EDSSs, enabling at the same time an easy integration of several complex technologies.

Within the same application segment (emergency management), the CHARADE project (CEC, 1995) can be seen as the predecessor of TRACE. While CHARADE was an experimental system focused on wildfire emergency management, TRACE is designed as a generic, cost-effective platform for the rapid development of applications in any environmental domain.

3. A Decision Support Framework

An important idea behind the TRACE project was to exploit research results in previous projects and to establish within a new research framework a convenient alternative to developing effective EDSS applications at a reduced cost.

As the name of the project suggests (TRAnsferring CHARADE technology in Central and Eastern Europe) the main objective of TRACE was to transfer to Central and Eastern Europe the technology developed within CHARADE, a project enacted inside the III Framework Program and evaluated as one of the 100 successful experiences of the ESPRIT Programme. In such a project, a generic platform for handling environmental emergencies had been developed.

In this respect, TRACE was intended to demonstrate the viability of transferring this technology both in new operational contexts and in other application domains.

The TRACE work-plan was organised in two main phases:

A) transferring two leading technologies of CHARADE, namely:

- *task-centred design methodology*: a user-centred design technique supported by a software tool for task analysis. Both the methodology and the tool provide a sound basis for the design and evaluation of human-computer interfaces;
- *a generic platform for EDSS:* a tailoring of the CHARADE platform including porting and down-sizing of basic modules.

B) development of two pilot applications demonstrating the viability of the transfer in new contexts and new application domains.

As a result of TRACE project a support for EDSSs development at two different levels was finally provided. On the one hand the support consists of a methodological framework including guidelines and tools for designing applications through task modelling methods. On the other hand a software library is provided consisting of a set of programmable objects, which implement basic functions for the management of environmental emergencies.

3.1 User-centred Design Through Task Analysis

It is broadly recognized today that a close connection between designers and users is an important premise for successful design. The designers should understand users as well as the technology for a successful design. Successful user-centred design happens when designers understand what users are trying to accomplish – the users' goals and tasks – and the users think about their tasks – the users' conceptual model of the work and tools.

Researchers in the field of user-centred design call the processes of interaction between designers and users as task analysis (TA) and task modelling (TM).

Task analysis and *task modelling* is a general methodology for modelling *activities* which can be observed in a large variety of domains from human factors (ergonomics and cognitive psychology) to artificial intelligence or software system design.

Task analysis contributes the software engineering process in providing an user-oriented view of the system, both at the conceptual and the detailed interaction levels.

The result of the TA is a task model, in which tasks to be system supported are fully described, including specification of each task (goal, description, objects involved, context of execution, task frequency and duration), and relationships among tasks (hierarchy, logical and temporal relationships).

TA includes two phases: pre-design and design. During pre-design, data on current practice are collected, by interrogating a certain number of potential system users. The pre-design task model thus contains only a description of user tasks. During design phase, the model is re-worked so that it describes the future activity the system under design is going to support. At this phase, tasks must be further analysed in order to distribute activities between the user and the system; to identify which user tasks are executed with the help of the system; to re-organize the activity in light of the novelties introduced by the system.

The design task model can thus contribute system design in several ways [Paggio, 1997]:

- it provides a structured view of user requirements;
- it supports the delimitation of system functionality;
- it provides contextual information for task execution, thus enabling effective design of the dialogue between the user and the system;
- it provides guidelines for task-based user validation of the system.

Many of DSS experts consider the user interface as the most important component of a DSS because much of the power, flexibility, and easy-to-use characteristics of DSS are derived from this component [Sprague et al, 1996] and

because the user sees only this part of the DSS, to him, the user interface *is* the system [Whitten et al, 1997].

In user-centred design, task analysis is a practical way of making sure that the user's perspective is adopted, and that the user interface is designed to support the user's actual work activities.

To be usable, an interface must also be perceived as usable by those who must use it or choose to use it. Usable interfaces have certain characteristics in common [Hackos & Redish, 1998]:

- They reflect the workflows that are familiar or comfortable
- They support the users' learning styles
- They are compatible in the users' working environment
- They encompass a design concept (a metaphor or idiom) that is familiar to the users.
- They have a consistency of presentation (lay-out, icons, interactions) that makes them appear reliable and easy -to -learn
- They use language and illustrations that are familiar to the users or easy-to- learn

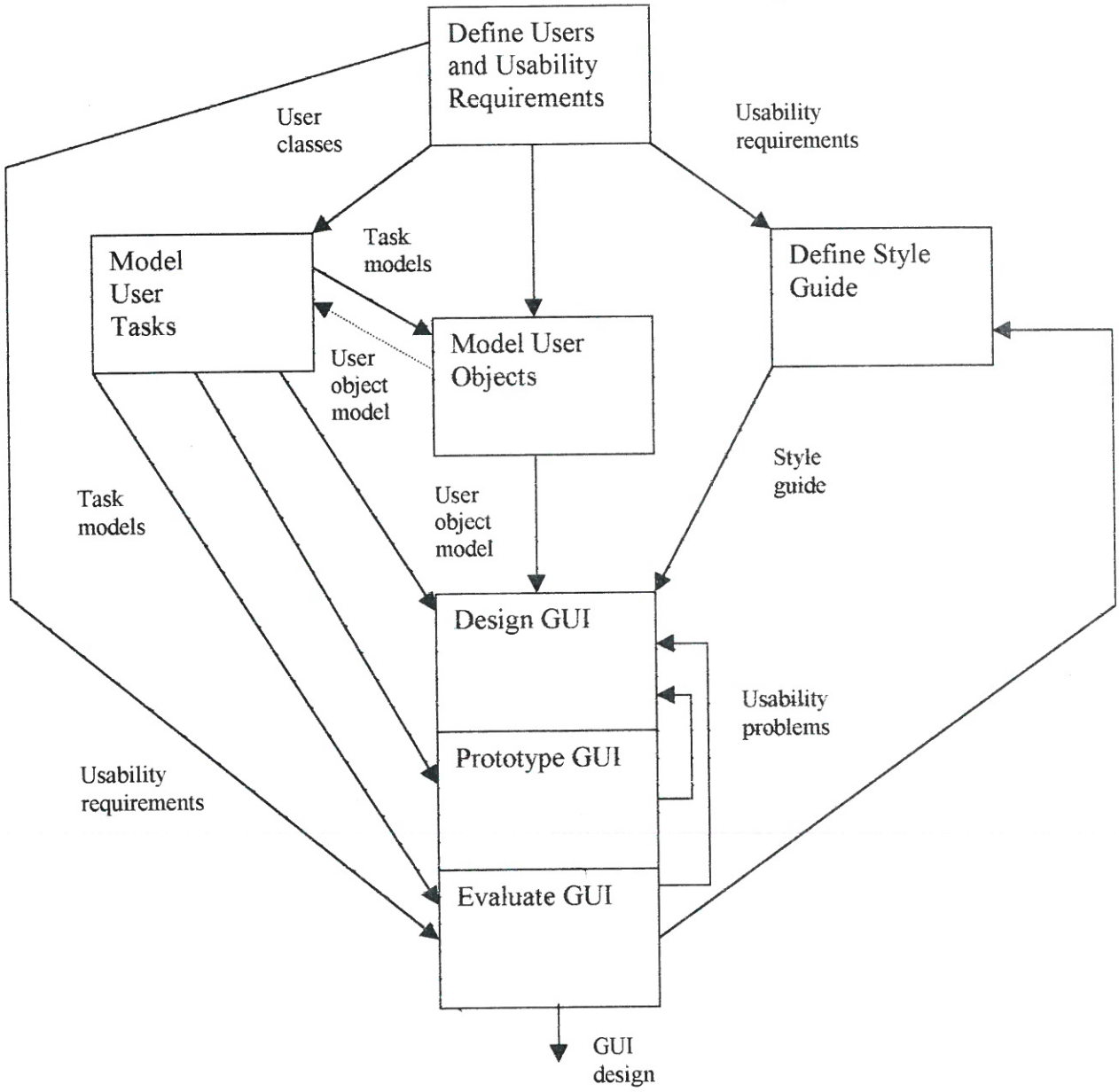

Figure 1. Overview of the Graphical User Interface Design and Evaluation Process

Figure 1 presents a high-level overview diagram of the graphical user -interface (GUI) design and evaluation [Redmond-Pyle and Moore, 1995] in the context of using task analysis as methodological framework. It is an idealized representation of the processes and of how they are related to each other. Boxes represent processes, and lines represent how products are produced by one process and input to another.

Various issues are considered before prototyping:

- end-users
- usability requirements
- user tasks
- user objects
- style guide

The processes of Design GUI, Prototype GUI, Evaluate GUI overlap and in practice merge into each other. The GUI design evolves through an on-going process of feedback from prototyping and evaluation. In fact all the processes are profoundly iterative. In the schematic overview diagram in Figure 1 the extent of this feedback and iteration is under-evaluated. All the earlier products – style guide, user objects, tasks, etc. – are liable to be revised as a result of the feedback from prototyping and evaluation.

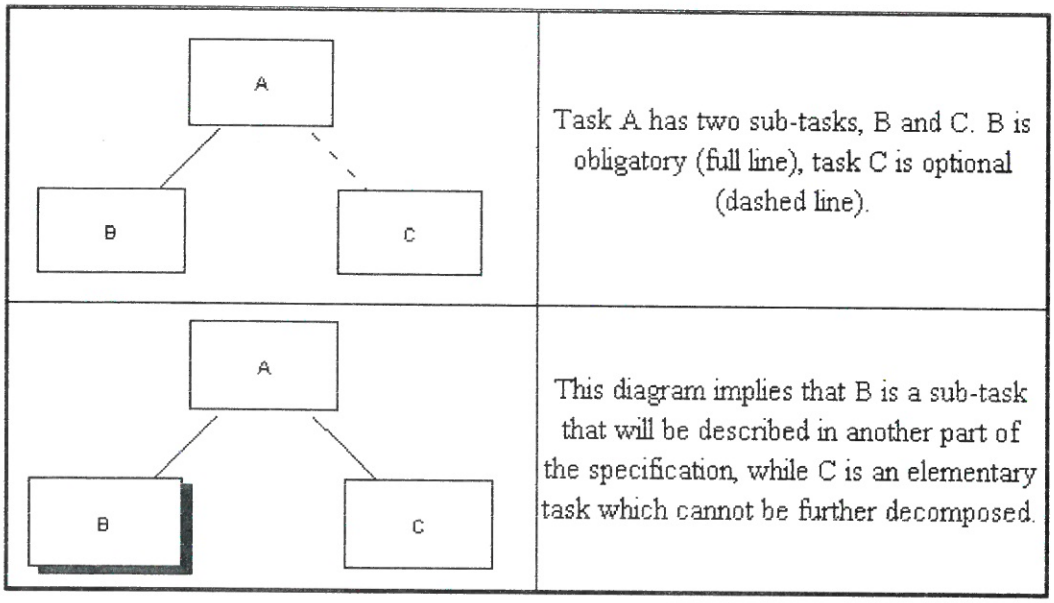

Figure 2. Representation of the Hierarchical Sub-task Structure

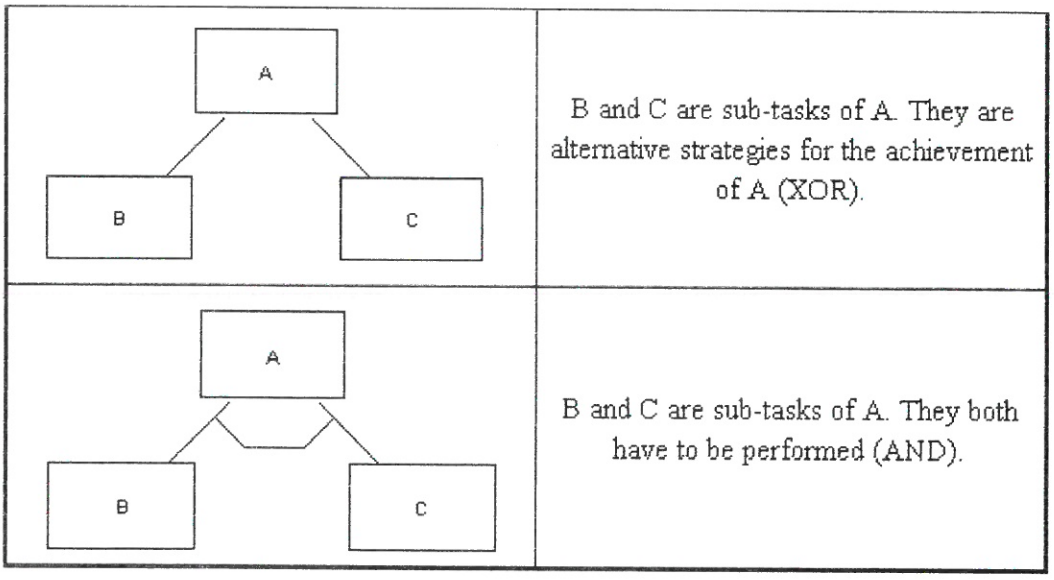

Figure 3. Representation of Logical Relations Between Tasks

3.2 Task-modelling Notation in TRACE

The task-modelling notation used in TRACE includes a graphical notation for representing tasks (which is the main kind of notation used in the task model) and a notation for representing objects. The graphical notation is partially derived from TKS notation [Johnson and Johnson, 1990]. The examples in Figures 2, 3 and 4 illustrate some of the means for representing hierarchical task-sub-task structure, as well as logical and temporal relations [Marti and Normand, 1995] [Paggio et al, 1997].

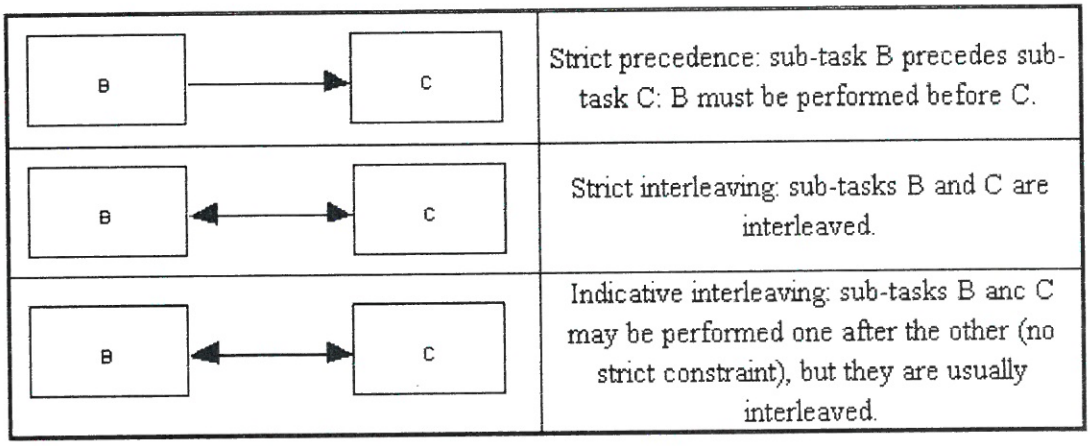

Figure 4. Representation of Temporal Relations

3.3 Tool for Supporting Task Modelling

Within the TRACE project was developed a graphical knowledge acquisition and documentation tool to support designers in creating and maintaining task models (Task Editor). The target environment of Task Editor (TE) is MS Windows 95/NT. The basic principles of TE design were the following [Paggio et al, 1999]:

- *Support conceptual level operation* by the user orientation on the concept of the task model besides its graphical representation. The interaction between the user and the editor is carried out mainly in terms of task analysis language, thus avoiding details which are irrelevant to the knowledge acquisition process.
- *Build syntactically consistent and complete models* by applying context-sensitive syntactical checks at each step of the process of creation of the task model and by maintaining strict correspondence between its textual and graphical counterparts.
- *Develop special means* capable to make it easier for the user to localise the incomplete fragments of the constructed model.
- *System driven interaction with the user* preventing the user from possible errors and allowing him to concentrate on the relevant part of the knowledge to be acquired.
- *Direct manipulation of the graphical structure* supporting graphical, palette-based editing capabilities for direct manipulation of task structures. This is an exploitation of the graphical tree metaphor, and a prerequisite for future extensions of the tool towards generality (configurable palette of task symbols and task connectors).
- *Unified processing of pre-design and design task models* facilitating operation with the editor.

With TE a domain model is represented as a document containing one pre-design and several design task models. Each design model is generated from the same pre-design model and reflects different aspects related to concrete software implementation. Each model contains three main parts - a set of task pages showing the task diagrams, the set of task model objects the tasks are referred to and the set of task pages currently marked for deletion. The latter denotes those tasks that are not currently included in the domain model but may be used in the future. All task pages are organized into hierarchical tree-like structures via hyperlinks used for splitting descriptions of complex tasks into several

manageable and completed chunks aimed at documentation. Each task page contains a task diagram representing a tree the nodes of which are tasks connected via temporal, logical, task-subtask or pre-/postrequisite task relations. Each task is in turn a complex object containing a number of attributes used for coding various types of declarative task knowledge.

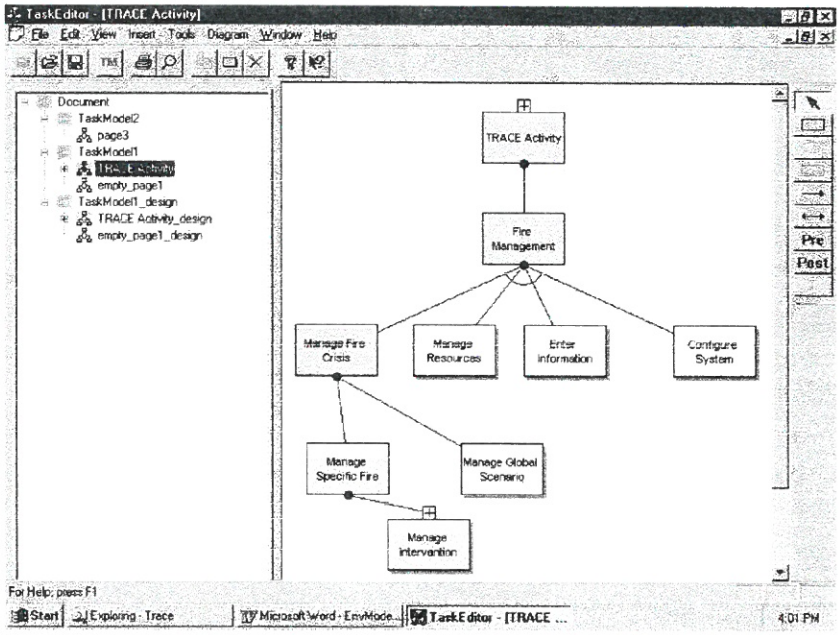

Figure 5. TE Screen Layout

The main TE screen is shown in Figure 5. The workspace is composed of a menubar, a message (status) bar, a graphical palette containing some graphical symbols used in the model and two scrollable panels: the *Structure Panel* and the *Task Diagram Panel*. The Structure Panel provides a synthetic view on the different task pages included in a task model, and typical tree control facilities for browsing and selecting task pages. On the diagram panel the active task page is presented. Task diagrams may be edited with direct manipulation from within such panel. The textual task attributes are created/edited by specialized dialogue boxes.

3.4 Representing Emergency Situations Through Task Models

Task models provide a convenient way of representing the decision-making process of emergency management. Information captured in task models may be exploited in a TRACE-based EDSS to guide the user in the actual decision-making process of each emergency management session. For each task in the task model the abstract notion of task *goal* is replaced by a set of concrete objects the value of which must be computed or updated by the system (or even provided by the user) during task execution. Thus a bridge between the functionality of the system and the user task model is established.

By implementing a propagation mechanism which automatically updates the realization status of each task - according to the status of computation of each object in the task's goal - we obtain a way of controlling the flow of activity for each EDSS session, and thus supporting the user in performing the right action at the right moment. This mechanism is exported at the level of TRACE-based application by one of the modules in the EDSS library developed within TRACE project and may also be applied to handle dependencies among computable objects.

An emergency situation may be represented as a network of tasks and situation indexes (see dependency management in the next Section) organized in a network of dependencies, which is ruled by a Subject-Observer mechanism [Paggio et al, 1999]. In few words said, the mechanism ensures that the status of each item – either a task or a situation index - is automatically updated during each phase of a session execution, preserving the global network consistency and enabling the user to monitor the current status of operations.

3.5 TRACE-based EDSS Architecture

Apart from Task Editor dedicated to support the knowledge acquisition and task modelling activity, within the project was also developed an EDSS library implementing basic and powerful functions for the management of environmental emergencies easy to integrate into EDSS applications. The library is implemented in C++. A general architecture of a TRACE-based application is presented in Figure 6. The target environment for the application built by using the EDSS library is Windows NT/95 environment.

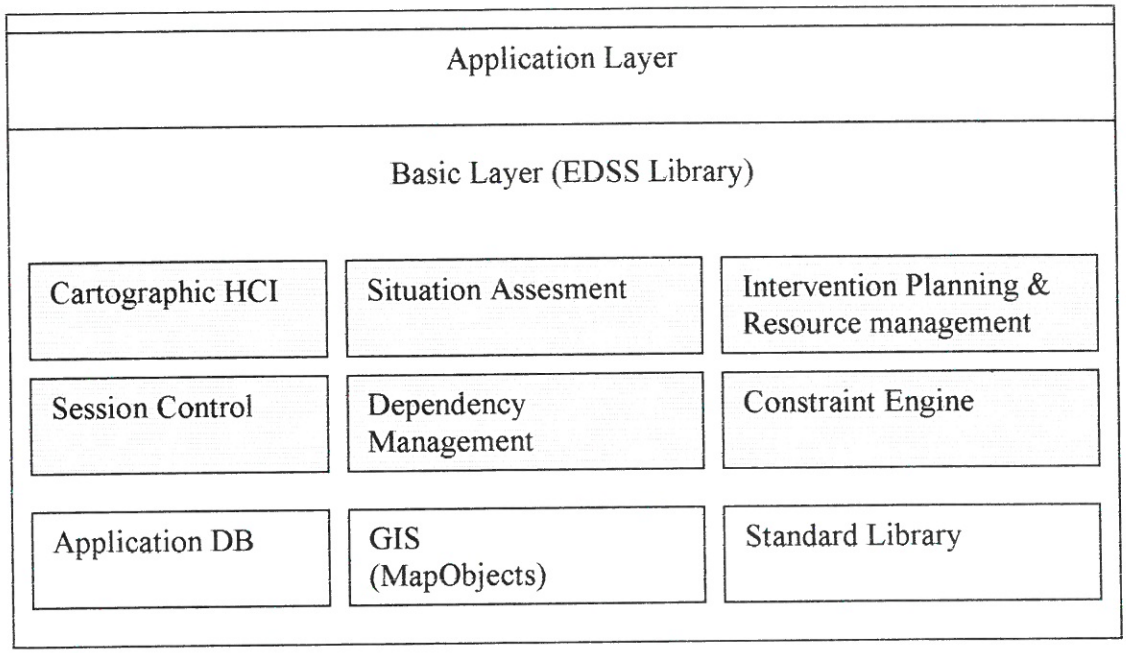

Figure 6. Main TRACE Based EDSS Components

The *Cartographic Interface* is in charge of driving the overall dialogue between the user and the system. It displays map elements and (georeferenced) application-dependent objects according to graphical conventions fixed by the user. From the implementation point of view the cartographic interface is linked to the Dependency management and Session Control modules. It exploits a run-time task model, i.e. a structured model of the activities of the user, and a session controller, which enables concurrent handling of multiple emergency situations. The task model is effective in driving user action throughout a session, by providing continuous feedback about the situation being monitored, showing the realization status of each task, enabling or disabling commands according to the current progress, displaying information which is relevant to the current task context.

The *Situation Assessment* module provides support for a quick analysis of an emergency situation, including basic spatial analysis and computational models based on geographically referenced data. It exploits a general mechanism for handling dependencies and propagating events among objects (see dependency management module), thus enabling automatic re-calculation of relevant information.

The *Intervention Planning & Resource management* module enables the user to define missions and automatically retrieve an appropriate set of resources to accomplish missions goals. This module has a very important role in a TRACE-based application dedicated to support the user in emergency management. The target user for such an application could be an operator of an emergency co-ordination centre, in charge of managing the available operational resources (means, equipment, personnel, and natural resources). The user is supported in assessing a crisis situation and building effective intervention plan. Requests for resources may be stated either in specific terms (e.g. a fixed number of resources belonging to a certain type), or as general criteria (e.g. minimization of intervention costs). User requests for resources can be expressed with a simple language by the user, and are then translated into constraint satisfaction problem, which is handled by the constraint engine module.

An *intervention plan* is a set of activities (*missions*) with the goal of facing an emergency situation. Plans are defined by the user on the basis of plan and mission stereotypes, i.e. predefined skeletons which must be filled with actual data referring to the situation at hand [Hayes-Roth and Hayes-Roth, 1979]. At mission level, different kinds of requirements for resources may be specified, namely:

- *general requirement* (latest start time for operations, selection of operational bases, accessibility of operations area);
- *type requirement* (request for specific types of resources);
- *job/capability requirement* (request for resources able to perform a certain job);
- *personnel requirement* (request for personnel, which is represented as a volumetric resource).

At plan level, global allocation criteria may be specified, which define the allocation strategy. An allocation criterion is a pair:

(<optimisation mode>, <variable to be optimised>)

whereas the optimisation mode expresses a preference on the values which can be assigned to the variable. The resource manager must associate each variable with an expression that computes its value; therefore a predefined set of legal variables must exist. For each actual mission, situation data must be provided, namely:

- the target area;
- the time schedule;
- the quantification of each requirement.

Resource data include for each unit both static information (type, owner base, set of quantified capabilities, average speed on different road types, amount of personnel that can be transported, availability status, usage cost) and dynamic information on the accessibility of the area. This dynamic information is supplied by algorithms that compute the shortest/fastest path on a road map from a resource base to the node which is the closest to the target point of intervention and it is stored in an appropriate data structure denoted mobilization table (see also the Application Scenario).

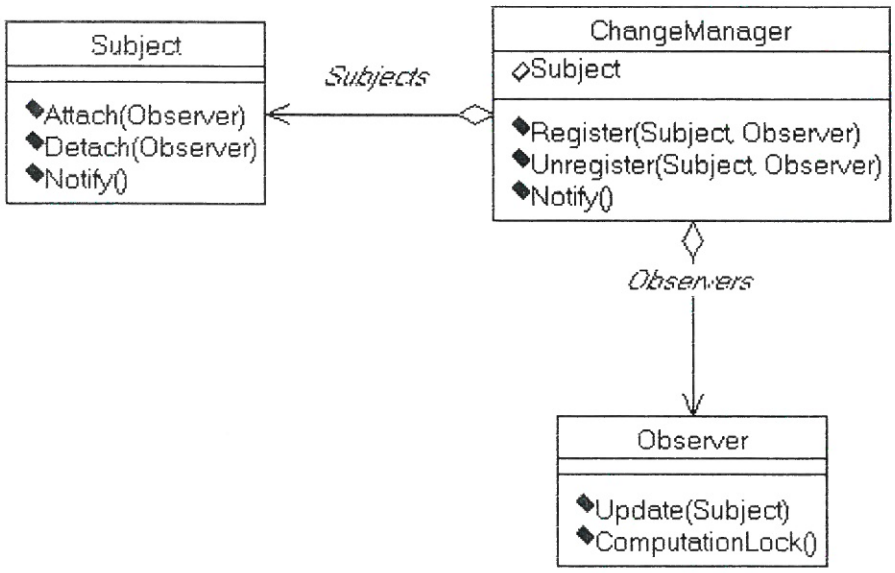

Figure 7. The UML Class Diagram of the Subject-Observer Pattern

The user-defined intervention plan translated into a constraint satisfaction problem is handled by the constraint engine module. When a solution is found by the constraint engine, it is fed back to the user in an appropriate form.

The main task of the *Constraint Engine* (CE) module is to handle user-defined requests, translated into constraint satisfaction problems by the Resource Management module. Beside being a component of the TRACE Library, the CR module can be considered as a general-purpose constraint solver, accessible as a C++ library.

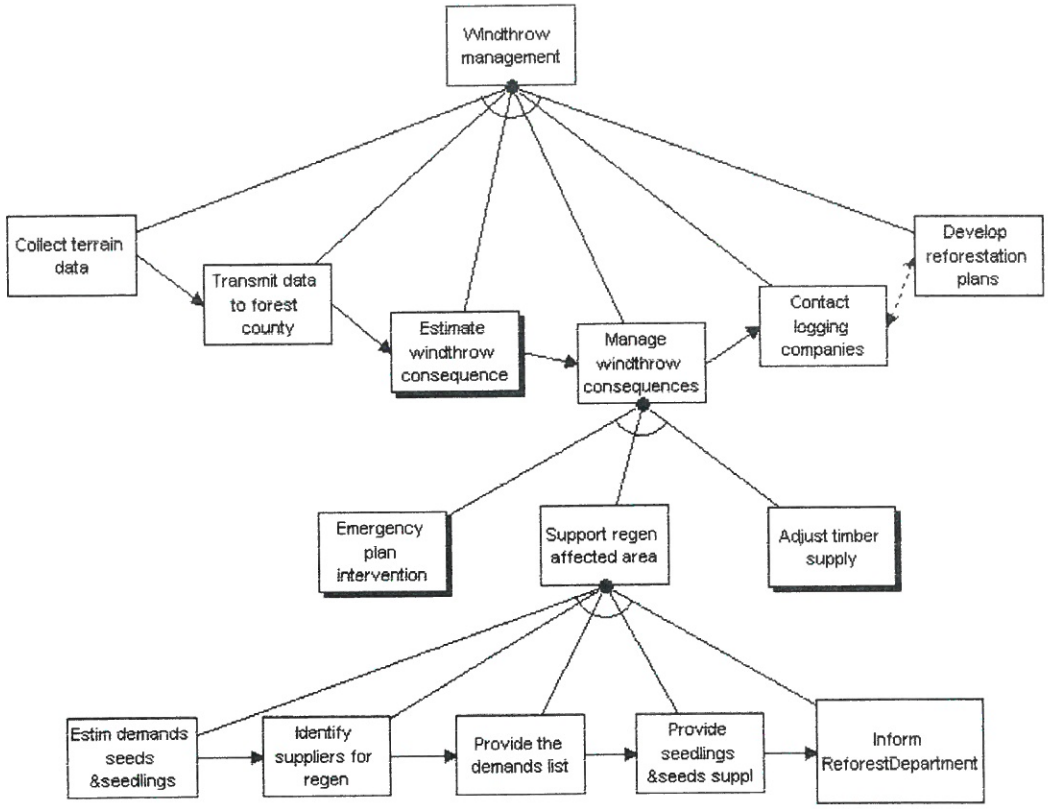

Figure 8. Windthrow Management Pre-design Task Model (Partial View)

The CE module is a finite domain constraint solver, using an object-oriented approach. In this sense it shows many similarities with the ILOG Solver [Puget, 1994], but also takes the special needs of the TRACE Library into consideration.

Dependency Management and its elementary dependency items (also denoted indexes in the context of TRACE) are exploited to model a number of heterogeneous entities and to make explicit their functional relationship through a network of dependencies. There are two main different classes of indexes in the TRACE context: task objects and situation assessment variables. In order to describe and assess a situation and its evolution, a lot of heterogeneous information, static or dynamic in nature, may need to be manipulated through different quantitative or non-quantitative models. Heuristics, rules or fuzzy sets could also be used to provide to the decision-maker with a synthetic picture of the situation. All these provide various variables that can be modelled as indexes. Examples of situation assessment variables would be: the global severity of the event to be controlled or its consequences, the entities at risk (people or values), the weather and time of the day, the suggested intervention strategy, suitable resources for intervention and so forth. Each index has a value the computation of which may depend on the values of other indexes. Thus an environmental specific emergency situation may be represented as a network of dependencies of tasks and situation indexes. A Subject-Observer Pattern [Paggio et al, 1999] to model one-to-many dependencies with automatic notification/update is exploited to maintain the consistency of the network. The main principles that were considered at the implementation of this mechanism were the following:

- abstract and minimal coupling between Subject and Observer,
- automatic broadcast notification to all interested objects that subscribed to it,
- exploitation of a single generic Observer update protocol.

In order to avoid needless and intermediate re-computations an appropriate mechanism was implemented in which related responsibilities have been distributed to both Subject and Observers. The notification can be selectively fired according with the status change and the update can be inhibited on the Observer side.

The pattern was finally exploited to maintain the system representation (task model) of the

flow of user activity up-to-date, since the status of a particular task depends on the status of related sub-tasks and also preceding tasks.

4. Application Scenario

In order to demonstrate the TRACE approach in developing EDSSs, two pilot applications were considered. The first EDSS application, denoted *Demonstrator 1* within the project (also called RO-TRACE), is intended to support the decision making in the case of windthrow management in Romanian forests. The second one, denoted *Demonstrator 2* is intended to support the decision making in the case of fire prevention and fighting in Bulgarian hard-to-reach massifs. The applications address two types of emergencies, frequently encountered in the European forest sector. The windthrow is a typical event mainly in the Norway spruce. The forest fires are quite frequent in the Mediterranean countries. The approaches in the two cases are different in many respects and only few features are common. While a fire should be controlled in real time and the most effort is done during the fire, in the case of a windthrow the main logistic problems occur after the catastrophic event has been finished. The differences concerning the type of decisions to be supported in the two cases and the intervention plans defined in both situations were taken into account in elaborating the specifications for the two EDSS applications.

The development of *Demonstrator 1* is based on the guidelines of the methodological framework proposed in the TRACE project. The process of knowledge acquisition and documentation has been following the task analysis methodology and has been implemented by the TRACE Task Editor. This was the first real application of that tool in practice.

Starting from the pre-design task model (Figure 8) that reflects the user view concerning the flow of activities in post windthrow management, we created the design task model (see Figure 9). It was obtained by evolving the pre-design task model taking into account what the TRACE system can do (the basic functions of the components of the generic software platform developed within the project) and

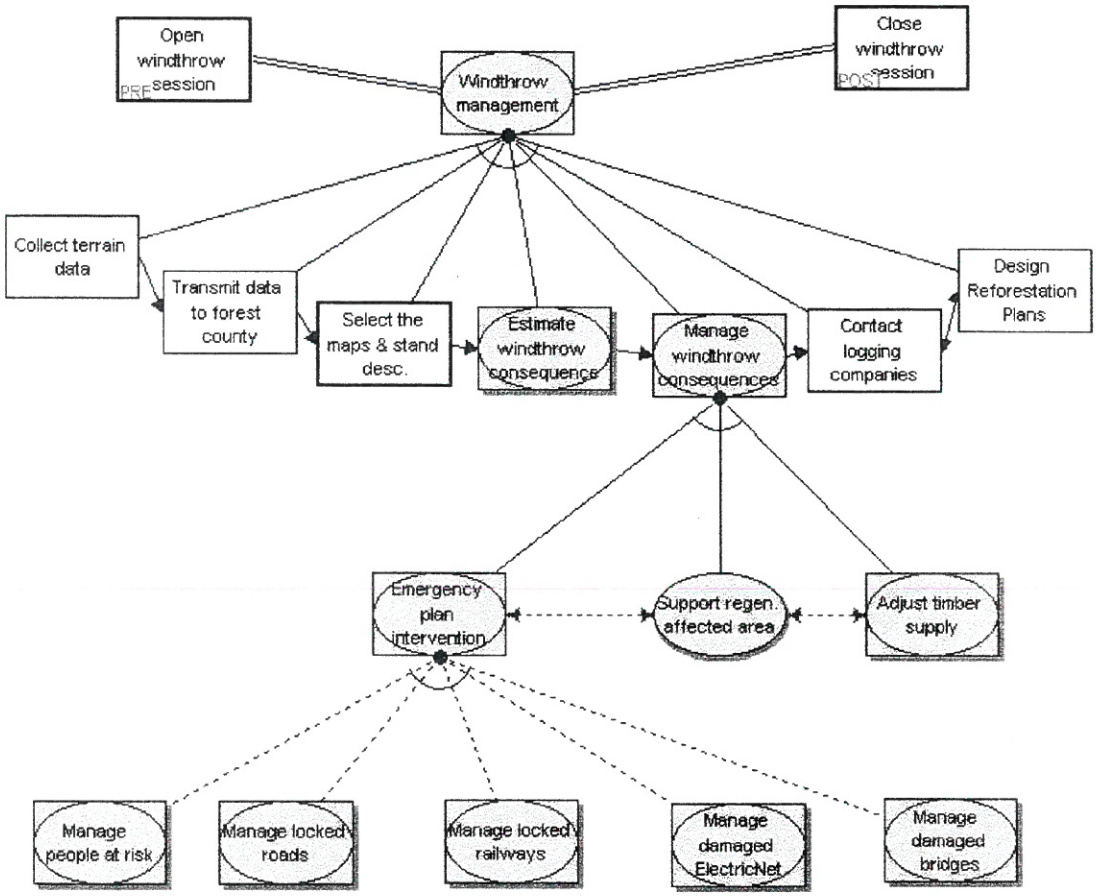

Figure 9. Windthrow Management Design Task Model (Partial View)

iterating either of the following:

- add new tasks or sub-tasks;
- modify existing task (by replacing simple tasks by more complex sub-trees) in order to separate system and user pieces of activity;

After allocating the tasks between the user and the system, the work was concentrated on the system tasks in order to identify the objects involved in the task execution. In addition to the domain task analysis mentioned above, the user data files available for running the application, the available cartographic data and in general all the data the users usually manipulate in case of a windthrow event were taken into account.

Apart from defining the application specific data, the types of the resources and their typical capabilities used in post windthrow emergency actions have been identified. In order to exploit the framework of defining actual missions in the TRACE system, a set of specific mission stereotypes to be integrated in the mission types library has been defined.

We identified also geo-referenced data specific to the application (data describing forest stands, forest roads, electric networks crossing the forestland, etc.). These geo-referenced data are manipulated using the GIS tool integrated in the TRACE system (MapObjects).

Demonstrator 1 mainly aims at supporting the decision-making in:

- post -windthrow *emergency actions* like estimate windthrow consequences, rescuing people in danger, opening locked roads, opening locked railways, repairing damaged electric networks, repairing damaged bridges, etc.
- post -windthrow *medium/long run actions* like estimating the timber volume in the affected area, designing reforestation plans for the affected area, modifying the cutting plans at the level of subordinated production units, etc.

Facing with an emergency situation is a current activity in post-windthrow management. Developing applications for supporting the decision-maker in this kind of activity was a primary objective of the TRACE technology. The RO-TRACE application exploits the dependency management mechanism by representing an emergency situation as a network of tasks and situation indexes organized in a network of dependencies. From the implementation point of view, the interface of the application is linked to the dependency management module. This connection means that the interface provides support for mapping user commands onto tasks requests. The overall dialogue between the user and the system is guided through underlying task structure, in

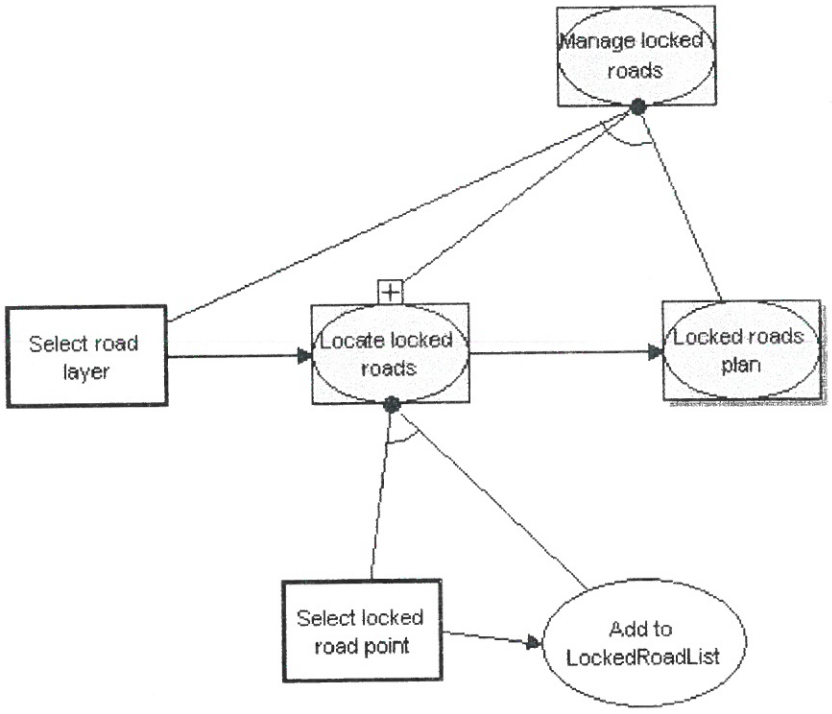

Figure 10. Manage_ locked_ roads Design Task Model (Partial View)

terms of tasks realization status, filtering of commands which are not enabled at a given stage of a situation handling, suggestions on the following steps to be performed.

The user is supported in rapidly assessing the emergency situation and creating effective intervention plans that exploits at best the available resources. Intervention plans are defined by the user on the basis of plan and mission stereotypes identified in the domain analysis stage. A simple example of mission stereotype with the goal of facing the emergency situation of locked road by fallen trees is presented below:

mission type name: open_road

a) resource types which might be involved in this kind of mission: **dozer**, **trailer**, **tractor**, **loading_tractor**, **sawing_machine**, **jeep**, **fuel_tank**, **sleeping_wagon**.
b) capability list includes: volume_to_remove, dozer_transport, trailer_transport, load/unload_logs, cuttings_logs, persons_carried, fuel_tank_transport, fuel_volume, workers_accommodation.

(**dozer** - *volume_to_remove*)
(**trailer** - *dozer_transport*)
(**tractor**- *trailer_transport, fuel_tank_transport*)
(**loading_tractor** - *load/unload logs*)
(**jeep** - *persons_carried*)
(**fuel_tank** - *fuel_volume*)
(**sleeping_wagon** - *workers_accommodation*)
(**sawing_machine** – *cutting_logs*)

In Figure 10 is presented the example of design task model for the management of this kind of emergency. The creating of the intervention plan is not detailed in the Figure, but when creating actual missions for the plan the user may start from the exemplified above mission stereotype.

The library of available mission stereotypes can be easily extended with new ones if the user considers necessary.

In the windthrow management session the overall allocation process starts from building mobilization table (MT), and is activated by the user by creating at least one mission, defining some requirements for it and launching the automatic allocation. In more detail, it may be decomposed into the following steps:

a. Build the mobilization table;
b. Define a user request for resources;
c. Apply the constraint reasoning techniques for selecting a suitable set of resources to be allocated;
d. Commit the solution (after the user confirmation).

The first step, automatically performed by the system after the access points in the emergency area have been identified, ensures the computation of dynamic data for each available vehicle in the resource bases. These dynamic data contain for each vehicle the shortest paths (or fastest paths, computed taking into account the average speed of every vehicle on different road types) from the owner base to the intervention points and the estimated travelling times (specific preparation time for each vehicle may be added).

In the second step the user defines his request following several stages:

1. Define a plan as a set of missions. The missions are defined from scratch or starting from mission stereotypes.
2. At the level of mission an appropriate control panel enables the user to declare the appropriate number of requested persons, the appropriate number of requested resources for each type and the amount required for each capability. This kind of declarations at the level of interface, represents a simple way for formulating requirement statements which in natural language could be expressed like:

   - "I want X amount of personnel"
   - "I want X resources of type Y"
   - "I want enough resources able to perform the X amount of the Y capability"

The user must specify also at the level of actual mission the deadline and the operational point (it must be one of the intervention points processed in the mobilization table).

Optionally the user has the possibility to specify general conditions for selecting the resources that may be used in mission. The conditions are specified using interface elements that enable the user to build and apply atomic filters or more complex logical AND-OR filters exploiting the filter generic classes provided at the level of Resource Management module. Examples of statements which can be managed by this simple formal language are:

- "I want resources being on emergency area before given deadline";

- "I want resources belonging to certain bases";
- "I want terrestrial resources from certain bases being on emergency area before given deadline", etc.

Applying the filters has the effect of removing from MT all those values which do not meet the user- defined statements.

3. At the level of plan the user may specify the global allocation criteria defining the allocation strategy. All the missions in a plan share the same global constraints. Appropriate interface elements enable the user to formulate in a simple way user preferences that in natural language could be expressed in statements like:

    - "I want to minimize intervention costs"
    - "I want to minimize the number of resources employed", etc.

Step c, performed by the system, ensures the translation of allocation problem into constraint satisfaction problem terms thus making it possible to exploit the capabilities of the constraint management module in solving it. If no solution is found the user may repeat the process from Step b, specifying a new, relaxed request.

Usually, for a given set of requirements, many possible solutions can be found. The user has the possibility to browse the solutions list and to select the preferred one. The commitment of the selected solution at Step d must not be done until the resource dispatching task is accomplished because resources that are marked "available" in the system repository might actually be not available, or not ready to be employed.

The decision-makers are interested in being supported not only in the emergency activities but also in the medium and long-term activities. The main medium/long-term activities are those concerning the regeneration of the affected area and timber supply adjustment.

Reforestation plans must be designed in the early stages of post windthrow management. The application provides useful data related to the species in the affected area and the genetic resources demand for each affected forest district. At the same time the application provides data concerning the seed and seedlings suppliers and their offer for the species in the affected area. Based on the information provided by the system, the users can decide on the appropriate policy for the regeneration of the affected area.

Adjustment of the timber supply is necessary in order to modify the current cutting plans at the level of production units so that they take the timber volume in the affected area into account. This modification is necessary in accordance with the *sustained yield principle* [Jobstl, 1997] which states that annual or decennial yields should be as even as possible. Cutting plans are modified depending on the volume of timber in the affected area and the implemented policy at the forest county level of decision. The EDSS application provides support for an appropriate analysis of the possibilities of propagating the economic consequences of the windthrow in time or/and in space. Propagating the economic consequences *in time* means adjusting only the cutting budget of the production units belonging to the affected forest districts for one year or more, depending on the size of the windthrow. The cutting budgets of the neighboring forest districts not affected by the windthrow, are not adjusted in this case. *In space* propagation means adjusting the cutting budget of both the affected forest districts and the neighbouring ones.

5. Conclusions

The framework for developing EDSS applications proposed in the paper is based on the research and development results obtained in the TRACE project in which 8 partners from 5 different countries were involved. The support in developing application is provided at two different levels. At the conceptual level the framework includes guidelines and tools for designing applications through a user-centred approach. At the level of application development an object-oriented software library is provided. The library consists of a set of programmable objects implementing basic functions for the management of environmental emergencies.

The main benefits to be expected from an application, developed following the TRACE approach are [Paggio et al, 1999]:

- to decrease the human error rate (time and risk pressure are key factors in emergency management, which result in error-prone decisions);
- to diminish the time required for performing a full assessment of the situation, by exploiting the Geographical Information System (GIS) visual capabilities and providing

the ability to filter and display context-driven information;

- to enable concurrent management of multiple situations;
- to support optimized and safe usage of intervention resources (means, personnel, equipment, natural resources), which are usually limited in number and capabilities, and therefore must be exploited at best;
- to facilitate the training of operators.

These features can be found also in other EDSSs but following the TRACE framework in developing applications they are provided in a cost-effective approach. The innovative character of TRACE approach is also related to the user-centred design it supports. The two applications developed within the same project following the framework proposed in the paper demonstrate the advantages of TRACE approach in developing EDSS applications.

Acknowledgment

This research was done with the support of TRACE INCO-COPERNICUS Project CP-960138, funded by the European Union within the ESPRIT Programme. The following partners were involved in this research and development project: *Italsoft (Rome, Italy-project co-ordinator), ICI (Bucharest, Romania), IQSOFT (Budapest, Hungary), Alenia (Rome, Italy), ISOMATIC (Sofia, Bulgaria), IIT (Sofia, Bulgaria), Otech (Paris, France), ICAS (Bucharest, Romania).*

REFERENCES

BATACHIA, L., CONSTANTINESCU, C., BOTAN, M. and DRAGOI, M., **Specifications of Demonstrator1 (RO-TRACE),** TRACE INCO-COPERNICUS Project CP-960130, Internal Report, 1998.

CEC-DGIII Industry, Applying Information Technology, 101 Success Stories from Esprit, 1995.

CHARADE CONSORTIUM, **Final Report**, CEC DGIII - ESPRIT No. 6095, 1996.

DHAR, V. and STEIN, R., **Intelligent Decision Support Methods**, PRENTICE HALL, 1997.

GUARISO, A. and WERTHNER, H., **Environmental Decision Support Systems**, ELLIS HORWOOD LIMITED, Chichester, 1989.

HACKOS, J. and REDISH, J., **User and Task Analysis for Interface Design,** JOHN WILEY & SONS, INC., 1998.

JOBSTL, H., **Dynamic Translation Model. A Concept and Tool for Forestry Planning and Validation**, Proceedings of the 4th International Symposium on Operational Research, Lubljana, Slovenia, 1997, pp. 273-279.

JOHNSON, P. and JOHNSON, H., **Designers Identified Requirements for Tools To Support Task Analysis**, Proceedings of INTERACT'90, 1990, pp. 256-264.

MARTI, P. and NORMAND, V., **Bridging Software Design and Usability Analysis Through Task Modelling**, in K. Varghese and S. Pflegel (Eds.) Human Comfort and Security, SPRINGER-VERLAG, Berlin, 1995, pp. 39-50.

PAGGIO, R., AGRE, G., DICHEV, C., UMANN, G., ROZMAN, T., BATACHIA, L. and STOCCHERO, M., **A Cost-effective Programmable Environment for Developing Environmental Decision Support Systems**, ENVIRONMENTAL MODELLING AND SOFTWARE 14, 1999, pp. 367-382.

PAGGIO, R. and DICHEV, C., **Task Editor User Manual,** TRACE INCO-COPERNICUS Project CP-960130, Internal Report, 1998.

PAGGIO, R., AGRE, G., DICHEV, C., MARKOV, Z. and DONCIULESCU, D., **Task Editor Specification,** TRACE INCO-COPERNICUS Project CP-960130, Internal Report, 1997.

PAGGIO, R. and STOCCHERO, M., **Specifications of the TRACE Basic Layer – EDSS Tools**, TRACE INCO-COPERNICUS Project CP-960130, Internal Report, 1998.

PUGET, J.-F., **A C++ Implementation of CLP**, Proceedings of the 2nd International Conference on Intelligent Systems, Singapore, 1994.

REDMON-PYLE, D. and MOORE, A., Graphical User Interface and Evaluation (GUIDE). A Practical Process, PRENTICE HALL, 1995.

SIMON, H., **The New Science of Management Decisions,** PRENTICE HALL, Englewood Cliffs, NJ, USA, 1977.

R. H. Sprague and H. J. Watson (Eds.) **Decision Support Systems**, PRENTICE HALL, Englewood Cliffs, NJ, USA, 1996.

TURBAN, E., **Decision Support Systems and Expert Systems**, PRENTICE HALL, Englewood Cliffs, NJ, USA, 1995.

WHITTEN, J. L and BENTLEY, L.B., **Systems Analysis and Design Methods**, Burr Ridge, IL, USA, 1997.

UML Notation Guide, Version 1.1, September 1997, http://www.rational.com/uml

http://nt2g.ici.ro/trace