HANDLING EMERGENCY MANAGEMENT TRAINING SCENARIOS: THE MUSTER SCENARIO MANAGER

Henning B. Andersen, Verner Andersen & Mads S. Larsen
Risoe National Laboratory
Systems Analysis Department
Human/Machine Interaction Group
4000 Roskilde, Denmark
Voice: 46 77 46 77
Fax: 46 75 51 70

ABSTRACT

The SCENARIO MANAGER is a module within the MUSTER system: a prototype computer system supporting the multi-user training of emergency managers and their staff in co-ordinating activities to cope with emergencies. The overall architecture of and requirements to the MUSTER system is described in the accompanying paper (V. Andersen & HB Andersen, 1995); see also accompanying papers by Balducelli et al. 1995, Hansen et al., 1995) The SCENARIO MANAGER has been designed to meet the requirements of training supervisors, that is, the persons (including aides) who plan, execute and analyse training sessions for emergency management trainees.

INTRODUCTION

The purpose of the SCENARIO MANAGER is to provide a flexible tool which supports the organisation and administration of training scenarios for emergency management training. In particular, the SCENARIO MANAGER is designed to support training supervisors in the three key stages of their tasks, namely the planning, execution and post-session analysis of training scenarios. The SCENARIO MANAGER is one of the modules of the MUSTER system but it may serve as a stand-alone system for scenario planning, while its role as supporting execution can be achieved only when it is linked up with other modules simulating target processes (toxic releases, fires etc.) and resource management. See the overview paper by Andersen & Andersen, 1995, which provides an introduction to the overall MUSTER system.

The objectives behind the design of the SCENARIO MANAGER are to provide support to training supervisors in accomplishing a range of tasks in handling complex training scenarios. There are two problems in particular which we have sought to address by designing the SCENARIO MANAGER. One problem is to do with the rigidity of even moderately complex training scenarios - this has to do with the degree to which the actual trajectory of events and actions during the unfolding of the scenario is determined by the decisions of the trainees. The second problem concerns a problem for supervisors in maintaining an overview of options and choice points during his planning of a training scenario and, even more critically, during its execution.

When the MUSTER project was planned, one main goal set out initially was to provide a flexible multi-user training environment which would allow trainees an opportunity to train their skills against realistic simulations of the types of physical (and social) events with which they need to cope during real incidents or disasters. Thus, the MUSTER system seeks to replace, for the relatively narrow purpose of training decision makers, large-scale field exercises during which trainees may command the deployment of units who cope, not with computer simulations, but with more or less realistic fires, the rescuing of 'wounded people' and other events that are included in physically simulated emergencies. There are, of course, several well-tested and in many respect excellent alternatives to field exercises when focusing on training co-ordination and decision making. The most frequently used method is to use some form of table-top tactical trainer environment which provides a 'mini-world' played out on a table-top and where the virtual emergency scenario is played out by physically simulating deployment of resources and movement of 'risk objects' (wounded people; people at risk, hazardous materials etc.). See in particular Haurum 1995.

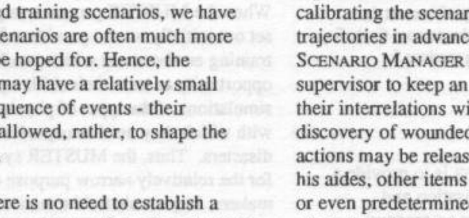
In contrast, the approach chosen in the design of the MUSTER system has been to allow trainees to cope with computer simulations of events and situations - and although we do not wish to criticise the often excellent training practices and effects observed in connection with table-top tactical training, we believe that computer simulation of events and processes can provide added realism, or at least different aspects of realism, to the training. Given, therefore, the goal of creating a flexible training system that allows trainees to cope with computer simulated events and situations, the problem of managing and keeping an overview of a scenario that involves interactive simulations becomes acute. Consider, in contrast, a large-scale field exercise. Since the logistics of such a scenario demands it that the sequence of nearly all main events are planned well in advance, the scenario will often appear, and will often be, rather more deterministic than would be the case in the real world. That is to say, since many or most of the major events have to be planned in advance, the decisions of the trainees, including their co-ordination and communication, will tend to have less of a causal influence on events than would be the case in the real world. Indeed, having interviewed a number of training experts and supervisors at various training schools (military, civil protection, firebrigades, railway emergency management, radiation protection) and having surveyed associated training scenarios, we have concluded that training scenarios are often much more deterministic than could be hoped for. Hence, the decisions by key trainees may have a relatively small influence on the actual sequence of events - their decisions and actions are allowed, rather, to shape the form and "size" of events.

In a table-top exercise, there is no need to establish a logistics planning at the same degree of detail and depth; the training supervisor and his aides may flexibly respond, by running the physical simulation of events by way of movements and indications on the table-top, and by providing plausible and non-deterministic responses to the decisions, commands and communications made by trainees. The only major drawback, it appears, is that trainees may lose the illusion that they are coping with an 'objective' situation rather than coping with the intentions and plans of the training supervisor.

The approach of the MUSTER system is therefore to allow a high degree of free choice and causal influence by trainees on the virtual events. This, however, does not mean that a training session necessarily requires less planning - it may do so, but that depends on the training goals and the type of scenario chosen. In general, to set up a training scenario requires that the training supervisor has drawn up in advance the main choice points during the scenario. Since the MUSTER system involves computer simulations of physical processes it is expected that a training supervisor who uses the system will, during the planning phase, run several mini-simulations in order to calculate critical parameters so as to make the scenario challenging but not impossible to cope with (i.e., he may calculate, say, the distance he wants to have between his initial fire and a neighbouring fuel tank allowing trainees sufficient time to cool the tank, if they act quickly). Having allowed trainees the liberty to interact with simulations, rather than improvised decisions by the supervisor and his aides, the supervisor has therefore a greater need for planning and calibrating the scenario and its different possible trajectories in advance. Hence, the design of the SCENARIO MANAGER seeks to satisfy the need of the supervisor to keep an overview of the 'choice points' and their interrelations with other outcomes (e.g., explosions, discovery of wounded people). Some of these events and actions may be released or initiated by the supervisor and his aides, other items may be released by the simulations or even predetermined parameter changes (change of wind direction etc.)

Underlying logic

The underlying temporal representational system on which the design of the SCENARIO MANAGER has been based is branching time logic. A branching time logic will contain a designated, perpetually moving NOW,



and every event that lies 'before' the designated 'now' is fixed and unchangeable and is located timewise on a single unbroken line stretching backwards. When we write it down on paper we usually go from left to right, so events in the past lie to the left of the designated 'now'. whereas events 'after' the designated NOW branch out into different futures.

In logic, the principle according to which a new branch is introduced (a 'choice point') is as liberal as possible: if the wing of the butterfly goes up, the future goes in one direction; and if it goes down, the future goes in another direction. For practical purposes we do not want to be nearly so liberal in acknowledging a new scenario trajectory: only "important" differences make a difference sufficient to warrant an explicit bifurcation of the training scenario. In this connection, a training supervisor (TS) will often introduce what we may call *alignments*. An *alignment* is a response by the TS or the training environments which takes into account that the trainees may do A or B and which brings the scenario back to the course laid down by the TS. So while A normally would lead to A*, say, and B to B*, a TS may decide that B* is much more valuable for the training session and he will therefore in the planning phase (or at the spur of the moment) introduce a perturbation so that even though trainees chose A and not B, the course of events nevertheless develop into course B*.

When we create a tree in which nodes represent important 'choice points' or important alternatives determined by events or non-trainee actions we shall speak of a *Scenario Tree*, see example below..

Implementation

In the following we describe only very briefly the implementation of the Scenario Manager, including a characterisation of its associated data structure and the principles behind the design of the *Scenario Tree* window. For a more complete description, please confer Larsen et al. (1995).

Terminology

In the Scenario tree we have *Events, Actions* and *Notes*. An *Event* is shown as a box in the main scenario tree and consists of a number of actions. An *Action* is the actual decision, action or event that make up part of the Event to which the action is attached. Actions are shown in two windows at the bottom of the Scenario Tree workspace, and describe the events in further detail. Finally, *Notes* are small reminders and comments. These can be owned by any external unit, and can be timed to any time in the scenario. In other words, in the railway scenario used as the demonstration scenario for the Danish version of the MUSTER prototype, there may be a set of notes for supervisor as well as a set of notes for the remote control centre. Notes may be connected to events. In Balducelli et al. 1995, the concept of Events and Actions is also referred to as 'nodes', conveying well the logical notion of choice points in a tree-like structure.

Selected technical features

The Scenario Tree Handler is able to draw a scenario tree on the basis of the underlying data supplied by the Training Supervisor when setting up the scenario script (the details of how the handler automatically generates a tree is described in Larsen et al. 1995). The Scenario Tree Handler must not only be able to draw a scenario tree but must also be able to animate the movement and changes to the tree during the execution of the scenario. Whenever an event in the tree reaches current time ('now') it will either be invoked automatically, or it will wait for someone (the training supervisor or one of his assistants), to acknowledge the event. What happens depends on how the event is timed.

<u>Automatic invocation of actions:</u> The Scenario Tree Handler will support automatic invocation of actions. Thus when an action is acknowledged, a set of objects and states will be set in the Muster Database, thus actually performing the actions in the system.

<u>Automatic detection of externally invoked actions:</u> The Scenario Tree handler will not automatically detect whether a specific action has been invoked. It will, however be possible for other programs to change the status of single actions in an event, thereby making the tree, proceed in the scenario.

<u>Change of time-scale:</u> The Scenario Tree handler may support changing the timescale in the Tree-window. This way it will be possible to make a "temporal zoom". By using temporal zoom the user can get a better overview of the scenario.

<u>Change of Information Detail:</u> The Scenario Tree handler may support different levels of detail in the objects displayed. In this way it will be possible to show the objects in different ways, making it possible to select a suitable balance between information detail and information overview.

The Overall Application Workspace

In the present paper there is no space for describing the technical details in even superficial form of the application - please confer the report by Larsen et al. (1995) for a complete description.

The Scenario Tree is presented in a stand-alone Visual Basic application. This application contains 4 windows, as outlined in figure 1. .

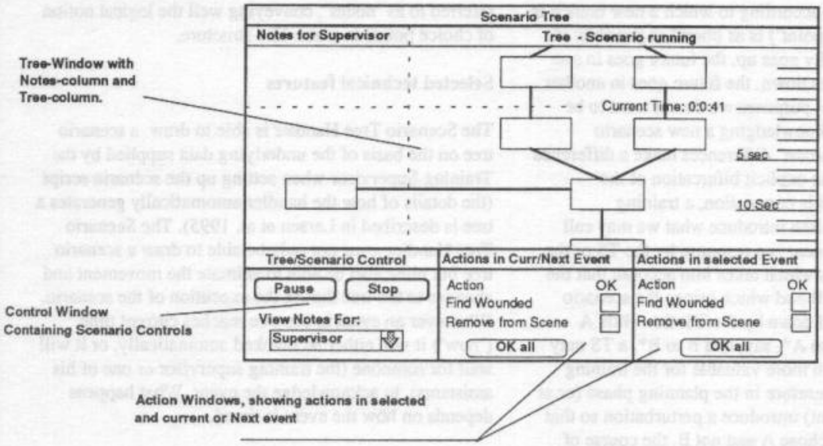


Figure 1: The Scenario Tree Workspace

REFERENCES

Andersen, V. and H.B. Andersen. 1995. "Multi-User System for Training and Evaluation of environmental Response - MUSTER". In Proc. *TIEMEC* 1995

Balducelli, C., S. Bologna, G. Di Costanzo, G. Vicoli, ENEA, M. Boero, AUTOMA: "A Computerized Support System to Cooperative Training in Emergency Scenarios Management". In Proc. *TIEMEC 1995*

Hansen, K. O., O. Østergaard and H.B. Andersen. 1995. "Emergency management training: Using a virtual reality environment to train on-site decision makers". In Proc. *TIEMEC 1995*

Haurum, G. 1995. "Tactical Training at the Danish National Fire College". In Proc. *TIEMEC* 1995

Larsen, M. S., H.B. Andersen and V. Andersen. 1995. "The Scenario Manager: A generic tool providing a self-generated training scenario tree". Risø National Laboratory, 4000 Roskilde, Denmark. Technical Report (R-Series).