An Event-Based Task Framework for Disaster Planning and Decision Support

Adriaan ter Mors, Jeroen Valk and Cees Witteveen
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology,
{a.w.termors, j.m.valk, c.witteveen}@ewi.tudelft.nl

ABSTRACT

Because of the apparent ineffectiveness of current disaster plans, we focus our research on modeling emergency response activities. If we can capture the crucial concepts of emergency response in a mathematical framework and apply this framework to construct disaster plans, then we pave the way for the development of automated decisions support systems for emergency response.

Keywords

disaster planning, decision support, events, task framework

INTRODUCTION

Many studies concerning disaster planning have found that actual operations in response to a crisis do not follow the instructions specified in the disaster plan. To gain better control over the emergency response operations, we should replace disaster plans in their current form – long, paper-based, and difficult to read – with disaster plans in a format that would allow more emergency responders to easily extract the information that is relevant for them. In our research, we have concerned ourselves with determining what the crucial concepts are in emergency response, and we are developing a mathematical framework in which these concepts can be modeled both clearly and intuitively. For one thing, we should be able to express which tasks should be performed, but also how one task depends on or is related to another, as dependencies between tasks will determine whether coordination and communication will be required to perform the tasks; furthermore, we should also be able to specify when and under which circumstances a particular task should be performed.

In our view, a disaster plan specified in a formal framework can form an integral part of an *automated decision support* system for emergency response – such a system could assist in deciding when a certain task should be performed, who should perform it, and which actions should be taken in case the actions planned in the disaster plan become inappropriate or impossible. Note that some researchers have suggested that disaster plans might be disbanded altogether in view of their ineffectiveness, and that research should focus instead on e.g. encouraging improvisation [5] and on improving communication between emergency responders [1]. However, we need an explicit disaster plan representation for the type of decision support system we envisage. Therefore, we view the research into e.g. improvisation and communications as complementary to our research.

Our research is still very much in the preliminary stages, so in the next section we will elaborate on our research goals that we have introduced above: to specify disaster plans in a formal framework, and to provide decision support using an information system based on this modeling framework. In the remainder of the paper, we will illustrate the use of our framework using a number of examples, but we do not present the framework in its entirety.

RESEARCH GOALS

Better disaster plans

We cannot give an objective answer to the question "what is a good disaster plan?" – and this issue hasn't been thoroughly analyzed as far as we know[1] – but we can highlight the shortcomings that we have come across in current disaster plans. We have studied disaster plans from the Dutch municipalities of Amsterdam (2003), Sevenum (2000), Volendam (1994), and Eindhoven (1993), and we have noted the following points:

- disaster plans are long, paper-based, and difficult to read;

[1] In [3], recommendations are given with regard to how disaster plans should be structured, and what information they should contain.

- information on one particular subject is dispersed throughout the document, some pieces of information are often repeated;
- it is unclear which parts of the disaster plan are meant for which emergency responders;
- how one activity depends on another is often specified vaguely, and in very general terms.

When we consider the points above, it is not surprising that studies have shown that only a handful of emergency responders are usually aware of the existence of the disaster plan – let alone of its contents [2,6]. To remedy this problem, we recommend that a disaster plan format should be concise and easily understandable, so that the majority of the emergency responders for which the disaster plan is relevant can become familiar with its contents.

Execution monitoring and plan repair

In the aftermath of a disaster it often – if not always – turns out that the emergency response operations did not follow the instructions specified in the disaster plan. Naturally, we would like to make this observation earlier, namely when there is still time to do something about it. We envisage a system that, based on the formalized disaster plan it has stored internally, *monitors* the emergency response operations and can notify an emergency responder when a certain task must be performed.

Of course, deviations from the disaster plan are not just caused by forgetfulness on the part of emergency responders. Frequently, *contingencies* arise that disrupt normal execution of the disaster plan; for example, equipment can break down, fires can spread, buildings can collapse, etc. In general, emergency response is one area where you should expect the unexpected. In the face of these uncertain conditions, one inclination might be to abandon all notions of planning, and focus instead on providing ideal circumstances for improvisation [5]. Although recognizing the need for improvisation is a good idea, we feel it can be complemented by contingency planning.

The disaster plan can explicitly plan for a contingency by specifying that if a certain *event* occurs, then we should perform a certain set of tasks. Contingency planning can also be done more implicitly; for example, we can specify that a certain task can be performed in more than one way. If one way of performing the task fails, the other *method* can be tried instead. In this way, the disaster plan is made more robust.

A HIERARCHICAL TASK FRAMEWORK

One view on incident management – a static, structural view – is that we must perform a set of *tasks*: victims must be rescued, their wounds must be tended to, fires must be put out; the list goes on. In addition, the execution of one task may depend on the execution of another. For example, before ambulances can reach the disaster scene, we must first clear the roads of the worst debris. Another, more dynamic view on incident management is that emergency response situations are characterized by incoming events: buildings collapse, equipment breaks down, alarm calls come in, key emergency response personnel arrives at the scene, etc.

To model both static and dynamic aspects, we are developing a modeling framework that combines both concepts. We will omit a formal and comprehensive description of our framework in favor of a brief listing of its elements, followed by a number of examples that show how the framework can be used to model portions of disaster plans. The framework contains the following elements:

- The processes in the disaster plan are modeled as a set of tasks *T*.
- Between the tasks in *T*, there can exist *precedence constraints*, expressing that one task should be completed before another can start; in addition, we may specify that a task can be completed by performing a number of its *subtasks* – all of its subtasks if we want to define an AND relation between, and at least one of the subtasks when specifying an OR relation.
- *Events* can occur due to some uncontrolled change in the environment (*external* events), but also on the completion of a task. In fact, a task can have more than one *outcome*, and we can model each outcome as the event that occurs on that particular completion of the task.
- An event can *trigger* a task, by which we mean that after the event has occurred, then the task can be executed (it has been *enabled*); an event can also *disable* a task, after which the task can no longer be performed. If a task was already being executed at the time the disabling event occurs, then the task will be *suspended*, and it will remain suspended until it is triggered again by a different event.

The following two examples aim to give an idea how a formal modeling framework can be used in disaster plan modeling, and they illustrate the framework elements listed above.