Dynamic Information System for Modelling of Design Processes.

H.J. Diederiks[a] and R.J. van Staveren[b]

[a]Icim BV, Information systems for civil engineering and environment, P.O. Box 5809, 2280 HV RIJSWIJK, The Netherlands

[b]Ministry of Transport, Public Works and Water Management, Directorate-General for Public Works and Water Management, Civil Engineering Division, Structural- and Road Informatics, P.O. Box 285, 2270 AG VOORBURG, The Netherlands

Abstract

DINAMO is a dynamic Information System for Modelling of Design Processes. It is intended for use along with product models, data management systems and existing applications. In DINAMO a programming user can define processes. These processes are represented by graphs. The graphs are characterized by nodes and relations between nodes. Each node in a graph represents a task, and each relation can be restricted to conditions. So the way in which a process is actually been performed, that is the actual path to be evaluated through the graph, can depend on certain conditions. Processes and functions (= software modules) are available to the user as tasks. A consuming user can activate tasks; the DINAMO system regulates the dispatch of the tasks, conform the process and function definitions.

Tasks are collected on sheets; sheets are collected in a task box. A task box can be regarded as a certain environment, determined by the programming user. A consuming user can choose between the environments which are available at that moment.

With the DINAMO system software and process definitions can be re-used in a simple way.

1. INTRODUCTION

We can look at classical information systems in two ways: functions as subject or data as subject. However, functions and data are often greathly incorporated in each other, like spagghetti. A clear separation, necessary for modular programming and re-use of software, is not always present. This evokes two problems: the user meets great difficulties in understanding the working of the software, and the software maintainer meets great efforts maintaining and error repairing.

In order to simplify this situation the Dutch Ministry of Transport, Public Works and Water Management started different research projects, both in the field of functions and in the field of data.

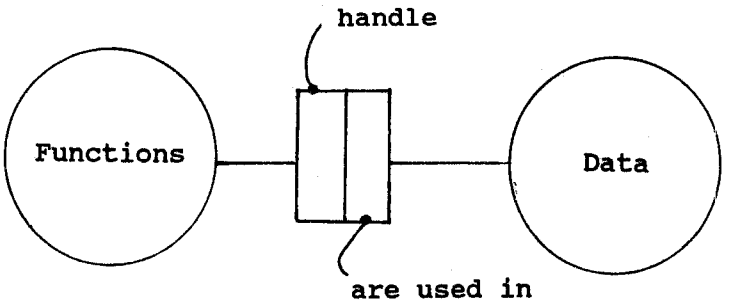

Figure 1. Subjects in conventional information systems

The projects in the field of data are concentrated on product models and the interchange of data. In the field of functions DINAMO was started, in autumn 1986. A look at the market showed that many firms had the same questions about dynamic process control, elementary functions and re-use of software, however an adequate solution was not offered. Therefore an own theory has been set up.

The DINAMO filosophy is based on tasks. Anything what the user wants to do or to be performed is called a task, no matter in which way it has to be done.

Tasks can be performed manually or by the computer. Tasks can be collected in processes. In the next chapters we will have a closer look at processes.

2. USER'S VIEW ON PROCESSES.

Everything what people are doing or want to do can be regarded as a process or as a part of another process. In this article we will concentrate on modelling, or design processes. Modelling can be regarded as: handling information, with a certain aim, that is to get a design. Thus the designer wants to perform tasks, on design data.

Example: a building consists of a certain number of "independent" parts. Foundation, bearing construction, walls, floors, roof, electrical installation, mechanical installations, piping, climate regulation. All those parts have to be designed, in independent and different processes, with great influence to each other, and handling partly the same objects.

Design data can be stored in product models. This is the right part of figure 1. Besides this, the designer must handle these data, by processes. This is the left part of figure 1. The total of processes we call a Process Model.

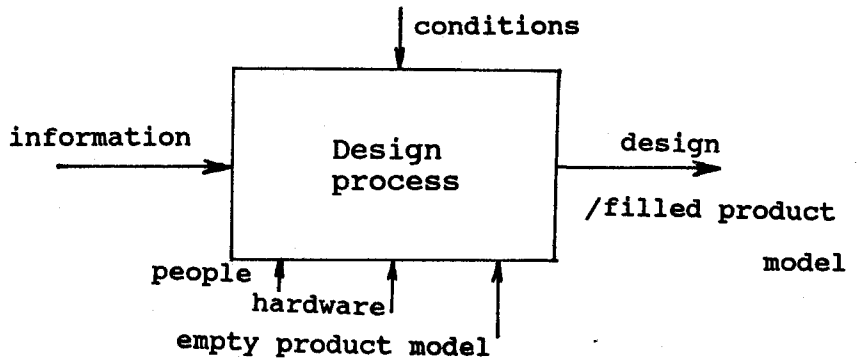

Figure 2. Design process and Product Models.

So, from the user's point of view, conceptual modelling is the total of process models and product models.

Product Models have to be filled and used, in all stages of the design process. It is irrelevant what kind of action it concerns: drafting, design, editing of text, or composing reports. Even the definition of product models can be regarded as one or more processes.

What about applications, in this context? Here we enter the system's view on processes. But first this:

Normally, the user only knows the existence of applications. About these applications the user has many concerns: what to use, when, how and where? All this information, by which the user is bothered, can be regarded as process knowledge, and can be saved in process definitions. Then the user can concentrate on his own and proper tasks, such as designing a building, and is released from "system tasks".

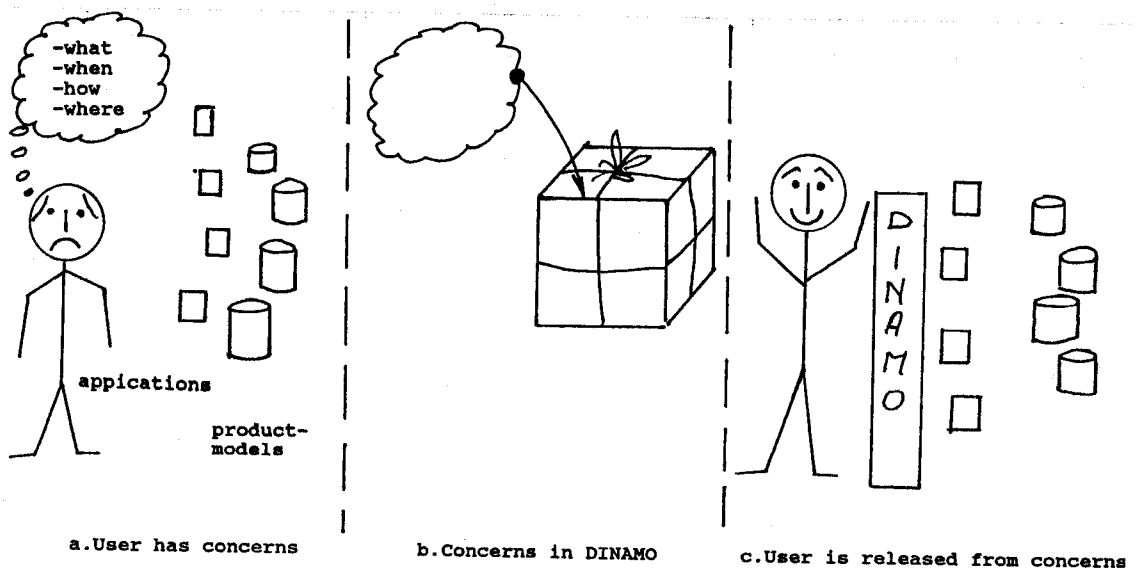

Figure 3. The user and DINAMO

3. SYSTEM'S VIEW ON PROCESSES.

For the system, a software module is nothing else than a task. It performs "something". The task is the functional unit, the software module is the technical solution for the task. In this case we say: the task is filled in by a function (task A in figure 4).

Tasks can be combined to a new task, a "master" task. We call the description of a combination of tasks a process definition. In this case we say: the task is filled in by a process (task A in figure 5). Task A contains information ("knowledge") about the combination of tasks B1, B2 and B3. It is therefore that we say that task A lies on a higher level.

In both cases, task A as a function and task A as a process, the task can be used in different higher processes, that is if the task is well defined, and the software module or process definition fits to the task.

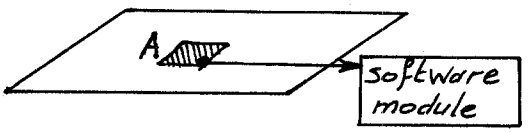

Figure 4. Task "A" as a function.

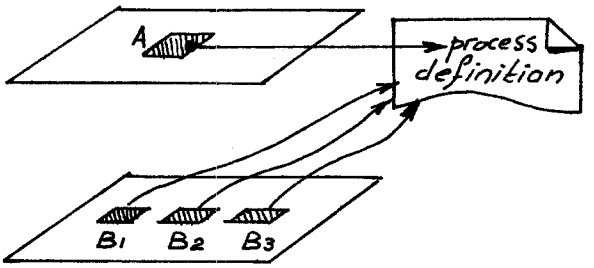

Figure 5. Task "A" as a process.

In DINAMO we do not use a computer language for the process definition, but graphs. They look like network plannings, with this restriction that in a planning planning all paths through the graph shall be performed, and that in a DINAMO graph only one path at a time can be performed, depending on the conditions.

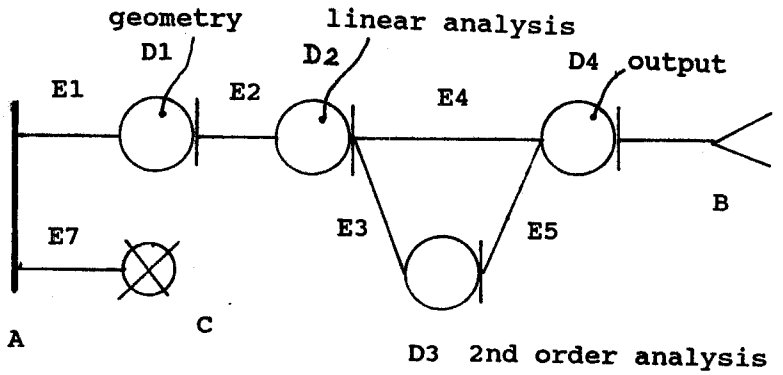

Figure 6. Example of a process definition.

A DINAMO graph contains five entities (see also figure 6): a starting point (A), an "end of path" (B), a stop node (C), normal nodes (D) and relations (E).

A normal node is the representation of a task. So the graph represents a collection of tasks, with relations between those tasks. A condition can be linked to a relation. If the condition during the evaluation is true, the relation will be valid. Example: in figure 6 task D3 (second order analysis) must be performed when the stress in a rod exceeds the yielding stress. Thus relation E4 has the following condition: rod stress < yield stress, and relation E5 has as condition: rod stress > yield stress.

In this example we see a special phenomenon. By placing tasks in a network, and describing conditions, the "programmer" adds knowledge to the process definition. The new task, to which the process definition belongs, lies on a higher level, on a higher sheet.

In the paragraphs above our "working direction" was bottom-up: composing tasks to new, higher tasks by process definitions. We can also work top-down. Let's regard task A in figure 5. Instead of a software module, as in figure 4, task A is filled-in by a process definition. Here the process definition is the technical solution for the functional unit (= task A).

The process definition has the shape of a graph, which contains tasks of one certain lower level. These (sub-)tasks can be filled in as a software module (an application, a message on the screen, a dummy function) or can also be decomposed in other (lower) tasks.

The network contains all the different ways in which the task can be performed. The way in which the task will be performed in reality depends on the conditions at the time of activating the task.

In both cases, bottom-up and top-down, we see levels. Levels, which are containing tasks. The tasks on the different levels can be regarded as independent tasks, or related to each other by process definitions. The collection of levels is called a "task box".

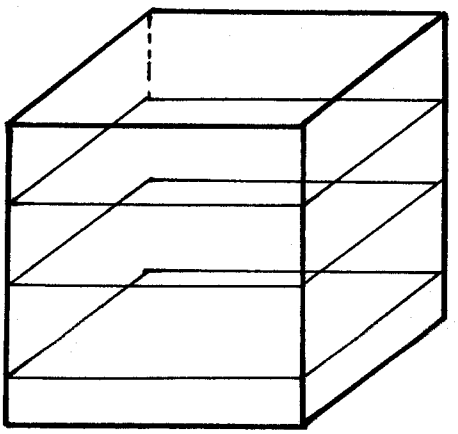

Figure 7. Task box

4. ASPECTS OF DINAMO PROCESSES

The DINAMO view on processes is not only a philosophy, but is also implemented in a tool-kit, called the DINAMO system. With the system on the background, the user can define or use its own processes. In this chapter we will regard some aspects.

- registrating and organizing design processes.
A typical characteristic of design processes is the different ways in which they can be performed, and the restriction to conditions. Both can be described by DINAMO graphs. So the graphs can contain knowledge about design processes.

- re-use of processes in other processes (layering).
The programming user can nest new and existing processes. A wheel has to be invented only once, and can be used as many times as the user wishes.

- minimizing source in the case of re-use.
Commonly, software is re-used by copying the source or retrieving it from a library. In DINAMO processes it's not necessary: software modules and process definitions stay on their own, represented by tasks.

- re-used software or processes stay independent.
Also, when a task is incorporated in high-level tasks by process definitions, the user can still approach the task as if it was a fully independent task.

- re-use over hardware-platforms.
The description of a function (see figure 4) can contain information like hardware nodes and directories, where the software module can be found. So the user is concerned with one process, while the software modules can be spread over different hardware platforms.

- customize these processes to own experience, needs and common use.
The (programming) user can easily change the process definitions by its own, without application obstacles and communication barriers. This possibility requires a special protection mechanism, in the form of user classification, with restrictions to features.

- modular programming.
The DINAMO graphs forces the programming user to pure modular programming. The programming user can make no calls-in-itself, so no recursion, or dependency from other modules. Only dependency from "outher" conditions is possible.

- process definitions are easy-to-read, and single-way-interpretable.
No knowledge of a computer language is needed, and on every hardware platform the graphs look the same.

- free task-object-coupling.
A task can be coupled with data, or objects (see figure 1). In case of a function, the objects concern input and output; in case of a process, the objects set the context of the underlying process and can be used for evaluation of the conditions in the

graph.

When the task is used in another process, the task can be coupled with other objects. See also figure 8.

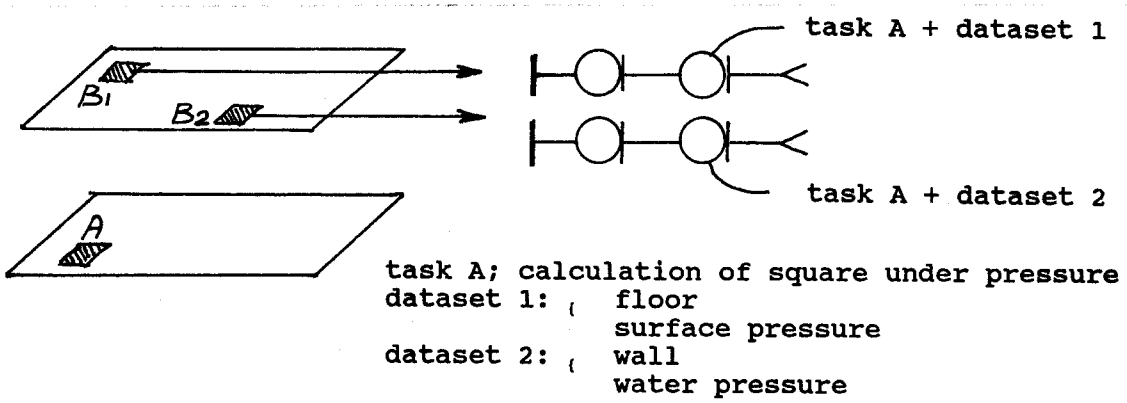

Figure 8. Free task-object-coupling.

- easy task repair.

When a process definition or function must be repaired, it does not affect the task to which it belongs, and so higher tasks are also not affected.

5. ASPECTS OF THE DINAMO SYSTEM

Besides a technique for registration of processes by graphs, DINAMO offers a system to handle those process definitions. Tasks can be defined and collected in a task box, and the tasks in a task box can be executed with the help of the DINAMO system.

Task boxes are interchangeable. The programming user defines what is in a task box and what is not. In other terms: he defines the environment. In figure 9 there are two task boxes: one for making product models, and one for architectural processes. The user can choose which task box will be used.

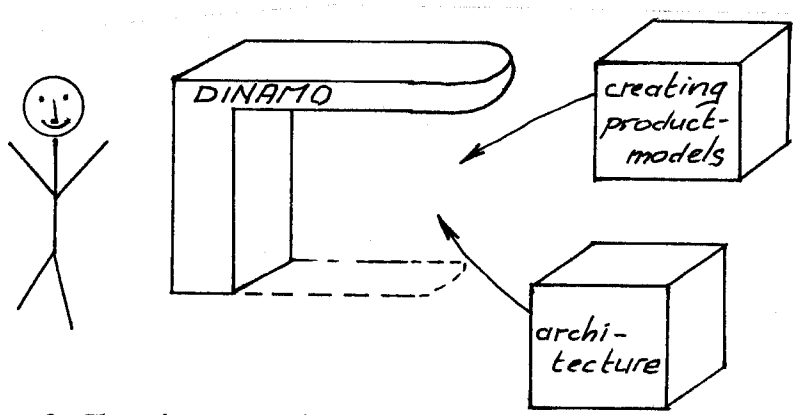

Figure 9. Choosing an environment (=task box)

DINAMO offers to the user four major system functions, collected in the Control Panel. See figure 10.

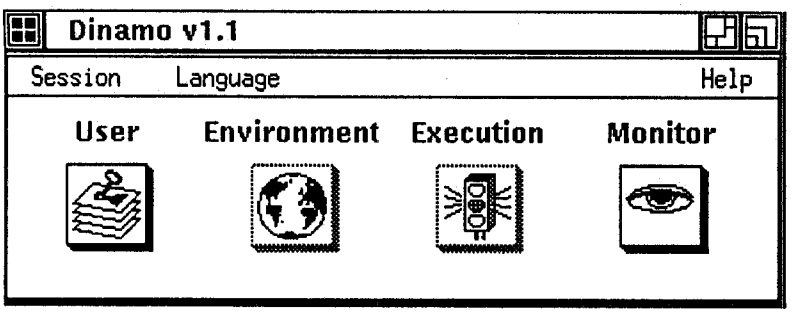

Figure 10. Control Panel.

The four system functions will shortly be discussed now.

- users:
Registration of DINAMO users. There are three types of users:
1. users who may define and change processes: programming users.
2. users who may use (activate) defined processes: consuming users. A consuming user can be restricted to certain tasks, for a special task box.
3. users who may use each task, and may change it: free users. Changes can only be made for own use.

- environments:
With this option the user (programming and consuming) can choose for a specific environment. After an environment is choosen, the user gets a sheet window, with one sheet of the choosen task box. See also figure 11. The programming user can define within the task box which sheet will be presented for which user.

- execution:
The user can influence the execution of processes. Options are: hold, resume, stop and restart.

- monitor:
Here the user can have a look "behind the curtains". Different options can be choosen. Output can be given on the screen or in a journal file. The information can concern the execution of processes, the communication within the DINAMO system

or the performance of the DINAMO system itself.

As mentioned under the system function "environment", after choosing for a special environment (= task box), the user finds himself on a sheet of the choosen task box. The sheet window gives a view on that sheet (see figure 11). In most cases it will be the upper sheet, but another sheet of the same box is possible.

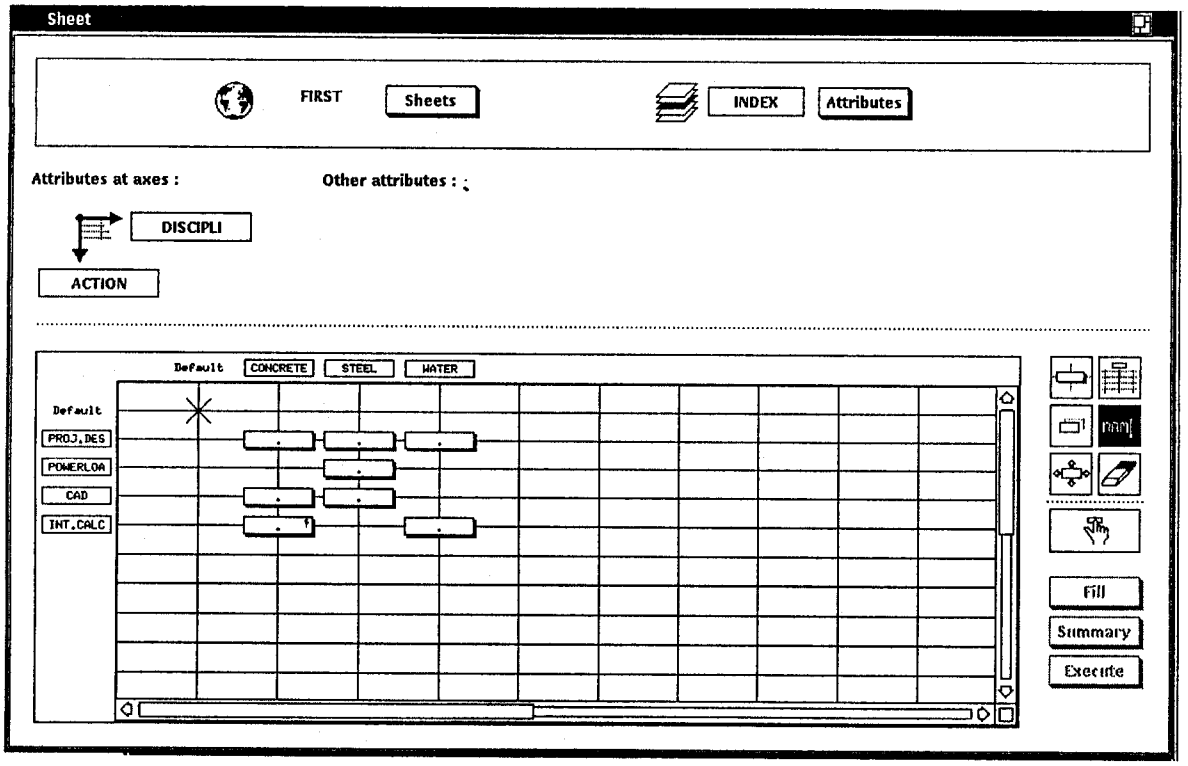

Figure 11. Sheet window

On the sheet, the defined tasks are visible. To help the user retrieve them, tasks are classified with the help of axes. A sheet can have two or more axes. Because of the 2D representation only two axes can be used for the representation. The system selects for default the first two axes, so the user always gets a picture. When the user wants another 2D view on a n-dimensional sheet, he can choose for another axis in the picture.

The meaning of a sheet depends on the type of user. A programming user will create, change and move tasks; a consuming user will only activate tasks. The sheet window only allows options which the actual user may use.

6. CONCLUSIONS

DINAMO is a system for registration, customizing and re-using processes, in combination with standard or company software (or mixed).

DINAMO offers flexibility, great reduction of software support efforts.
It simplifies the use of software by customers and offers possibilities for creating own processes in a simple way. Therefore DINAMO has an easy graphical interactive user interface, based on the international standard DecWindows. DINAMO gives the user a knife and a lunch-box, so that the user can compose his own sandwiches.

DINAMO is growing out of its research state. The first release will be ready in october this year.