A version of this paper has been published in: *Integrated Computer-Aided Engineering*, 9 (1), pp. 87-98, 2002.

Interpreted Petri Nets used for Human-Machine Dialogue Specification

F. MOUSSA*, M. RIAHI*, C. KOLSKI**, M. MOALLA*

** LIPP-Dép. Sciences de l'Informatique- Faculté des Sciences Campus Universitaire 1060 Le Belvédère- Tunis, Tunisie, Phone: (216 1) 872 600, Fax: (216 1) 885 190*
email : faouzi.moussa@ensi.rnu.tn, meriem.riahi@insat.rnu.tn

*** LAMIH - URA CNRS 8530 Université de Valenciennes et du Hainaut-Cambrésis - Le Mont Houy, F-59313 Valenciennes cedex 9, France, Phone: (33)3.27.51.14.69, Fax: (33)3.27.51.13.16*
email: kolski@univ-valenciennes.fr

Abstract: this paper proposes an approach for the specification of human-machine dialogue for interactive process control applications. This approach is based on a formal modelling of the Human-Machine System behaviour. This modelling makes it possible to deduce the user requirements and then to identify the User Interface (UI) objects. The formal aspect allows the validation of the specifications before going on to the generation of the interface.
A formalism using Interpreted Petri Nets is proposed for modelling the Human-Machine dialogue.

Key words: User Interface (UI), Human-Machine Dialogue, Human-Machine System (HMS), Formal Specification, Interpreted Petri Nets, Dialogue modelling

1. Introduction

Over the last decades and especially in the late nineties, we have seen on the one hand a considerable development in computer science, and on the other hand, a significant growth of industrial processes with a high degree of security such as chemical industries, nuclear processes and transport systems. In such fields, human errors can have dangerous consequences for the production system, the environment, for safety and human lives [41]. In order to avoid these types of human error, it was necessary to consider the problem at its root i.e. at the user interface (UI) design stage. Indeed, graphical interfaces represent the predominant way: (i) of informing the operators of the process evolution and (ii) of assisting them during their mental problem-solving tasks [39] [46] [21].

Considering the crucial role of the UI, the tendency in control rooms consists in presenting data on graphical screens. Powerful tools intended for graphical creation are very often proposed but misused [6] [26] [25]. The main problem is essentially methodological rather than practical. In fact, several ergonomic problems related to the use of the GUI are noticed. These problems mainly concern WHAT to present to the operators according to the functional context and the corresponding human tasks, and HOW to present the information graphically.

In this research work, we focus on the study of methodologies for UI design and development. As the studied area takes place in the industrial process with high degree of criticality, it is strongly recommended to involve formal tools for specification in the process design. We try, in particular, to overcome the following two problems: (i) the choice of rigorous formal techniques which are adequate for the application's context; (ii) the proposition of a global approach for UI design dealing with these formal techniques. This paper starts with a brief critical synthesis of related works on these two points. It then describes the proposed approach for UI design, focusing on the modelling of human-machine dialogue with a formal technique based on Interpreted Petri Nets.

2. Related Work

By analysing the approaches proposed in the literature on Human-Computer Interaction, we notice that most of them consider only one aspect of the interface design problem: task analysis, working with guidelines, interface specification, interface generation or evaluation [44]. We also notice that the tendency nowadays is to integrate formal techniques into the process of UI specification [35].

In fact, only formal techniques allow the designers to describe the external behaviour of any system without considering the implementation context in a way similar to XML [36]. These techniques also permit the validation of specifications before going on to the actual UI generation. They are especially recommended for complex and critical systems such as industrial processes [53]. So, it is very interesting to use a formal technique to specify a UI for process control. The problem, however, is deciding which technique is more adequate and suitable for UI specification.

In the literature, we find several formal specification techniques based on different theories: universal algebra, typed logic, theory of sets (Z, VDM, B) or automaton theory (state transition diagrams, Petri Nets, etc.). Each of these families has a preferred domain (data processing, real time, protocol management, etc.), a community of researchers and users and a number of variants or disciplines [27] [50].

In particularly, in the field of human computer interaction, various researchers have proposed formal methods. The state transition diagrams were proposed during the 60's, for modelling the screen chaining [37]. Jacob then proposed state transition diagrams for direct manipulation interfaces [18]. They were followed by Petri Nets and extensions [53][24][34], then temporal logic and LOTOS [38], the Z formalism [14], the Lustre language [7] and recently the B method [2]. Here, we present several of these approaches.

ICO formalism [34]: Palanque and Bastide propose modelling the interactive application's interface using the ICO formalism (Interactive Cooperative Objects) based on an object Petri net. This provides a semi-formal specification of the UI and its behaviour. However, the use of an object as token does not allow the validation of the Petri net model. Otherwise, before validating the model, it would seem to be necessary to give a detailed explanation of the conditions and the rules for transforming an object Petri net into an ordinary one on which validation is possible. This does not appear in this approach. Moreover, this method does not explain the principle of interactive object identification. It leaves that to the designer's common sense.

TOOD method [24] [49]: The TOOD method, proposes a design approach based on a hierarchical decomposition of the HMS into tasks modelled using Object Petri Nets. They build a structural task model from which they deduce an abstract interface model. An object design method of the UI is then proposed. But there are several problems which have not yet been solved. TOOD does not clarify how to specify the "graphical displays" and to generate them. Moreover, it does not consider the dysfunctioning modes in the HMS analysis so the task analysis is disconnected from the different functioning contexts. Like the ICO method, the validation of the object Petri net, in TOOD, needs further research.

XDM formalism [9]: A formalism called XDM (standing for conteXt sensitive Dialogue Modelling) is proposed by De Rosis and Pizzutilo for the formal description and evaluation of user interfaces. The formalism is based on extended Petri Nets and uses KLM operator theory (proposed by [5]). The formalism allows both the description of different static and dynamic aspects of the interaction in different functioning contexts, and the assessment of the interface. In fact, using this formalism, some properties of the UI such as correctness and usability can be easily or automatically verified. The method was developed within the framework of a medical system. The formalism allows the translation of the task analysis results towards the design of the interface. The Petri nets used allow the modelling of the human machine dialogue and the description of the system's states and user's actions. The KLM theory makes it possible to estimate the time necessary to establish an elementary action. For the general aim of dialogue modelling, XDM gives an appropriate method stressing the validation of the interface. However, it does not seem clear how the user requirements are constituted before their dispatching to the places. The choice of the graphical objects is left to the designer. Finally, in the XDM method it seems that the graphical object behaviour is not modelled as it is done for the HMS dialogue. We therefore wonder whether validating the interface at a high level of abstraction, without considering the object behaviour, is "strong enough"?

Approach proposed by D'ausbourg, Durrieu and Roche [7]: The purpose here is to describe the interactive behaviour of a system by building its abstract formal model and to verify automatically that this behaviour processes the required properties. They propose deriving a formal model, using the Lustre language, from the description of the intended interface as it was informally designed. The approach consists, therefore, in deducing a logical model from the Lustre model by using state charts [13]. The formal model is used here to verify properties of the system and not to specify and generate the interface. They made the assumption that the user or the designer builds the interface by using an interface generator tool, UIM/X, which generates a description of the interface with the UIL language [16]. They then derive a formal model from this description in order to verify and test the interface and its properties. This kind of model is useful for the evaluation of the interface and of its dialogue. It cannot be considered as a global methodology for UI design using formal models.

Approach based on the B Method [2]: These authors suggest using a well-defined formal method in interactive design context in their approach, without defining or using any interactive model. The B method is based on model description such as VDM or Z. These methods consist in defining a model using the variable attributes which characterise the described system, the invariant that must be satisfied and the different operations that alter these variables. The B method is based on predicate logic and on the weakest precondition calculation. It allows the support of specifications through abstract machines as well as refinement and implementation. With this method, it is possible to put the whole development in a common language, with a common semantics and a common proof technique. It therefore provides a uniform approach for the description of program developments. The approach consists in giving a model-independent method which can be customised to specific HCI models. The great strength of the B method is its capacity for refinement. That is the possibility to finally derive concrete programs from abstract specifications. However, the method does not explain how to constitute the user

requirements and how to decide on the UI objects. The effective generation of the graphical interface needs further research.
The main criticism of techniques such as B, lotos, or Lustre, concerns the relative difficulty in reading and manipulating their notations.

Ergo-Conceptor system [28] [30]: Concerning particular industrial applications, the Ergo-Conceptor system presents an ergonomic approach for UI design. This approach begins with a description of the user requirements into a database. By studying this database automatically, the system allows automatic UI generation using a knowledge-based approach. However, in this approach, there is no indication on how to identify the user requirements. It allows only a static interface generation. It did not consider the dynamic aspect of the interactive objects and the Human Machine dialogue.

The results of the bibliographical study are represented in Table 1. It shows that none of these approaches consider the crucial aspect of dysfunctioning analysis. The produced interfaces will therefore only be adapted to normal functioning situations. A dysfunctioning can, in this case, bring about a critical unpredictable situation. Almost all of the approaches discussed here consider the task analysis. Nevertheless, few of them explain how to integrate and to exploit the results of these analyses automatically in their models. Apart from the Ergo-Conceptor tool, none of the proposed approaches take into account the important aspect of the use of guidelines in the UI specification. This research line becomes particularly important since Tools for Working With Guidelines (TFWWG) are proposed increasingly frequently in the literature [12] [43] [51] [52]. These aspects will be given in more detail in paragraph 3.4.

Table 1. Evaluation of several approaches proposed for the interface design

	Palanque (ICO) [34]	***Mahfoudhi and collegues (TOOD) [24] [49]***	***De Rosis (XDM) [9]***	***D'ausbourg and al. [7]***	***Ait-Ameur and al. [2]***	***Moussa (ergo-Conceptor) [28] [30]***
Task analysis	*No*	*Yes*	*No*	*No*	*No*	*No*
Dysfunctioning analysis	*No*	*No*	*No*	*No*	*No*	*No*
Explain how to identify the user requirements	*No*	*Yes*	*No*	*No*	*No*	*No*
Consider the user requirements for the UI design	*Not precised*	*Yes*	*Yes*	*Not precised*	*No*	*Yes*
Consider the ergonomic criteria	*Not precised*	*Not precised*	*Not precised*	*Not precised*	*Not precised*	*Yes*
Use of a formal technique	*Yes (Petri Nets)*	*Yes (Petri Nets)*	*Yes (Petri Nets)*	*Yes (Lustre)*	*Yes (B)*	*No*
Provide a software tool	*In progress*	*In progress*	*Yes*	*No*	*Yes*	*Yes*
Validate the specifications	*Yes (Partially)*	*Yes (Partially)*	*Yes*	*Yes*	*Yes*	*No*
Generate the interface automatically	*Dialogue*	*Yes (Partially)*	*No*	*No*	*No*	*Graphical displays*
Provide a complete integrated approach	*No*	*Perspective*	*No*	*No*	*No*	*No*

Our aim is therefore to overcome these shortcomings and to propose a global approach for UI design and automated generation in process control. The implementation of this approach leads to the Ergo-Conceptor+ tool. The major points to cover are summarised below:

- Take the operator task analysis into account [19]. Because the operator's informational needs vary according to the functional context and the level of difficulty of the task.

- Identify the user requirements from both the task analysis and the state of the system. Thus, the information presented to the operator will be adequate and suitable.

- Deduce the graphical interface's objects. This is done automatically according to the user requirements.

- Use of computerized tools for working with guidelines to specify graphically the deduced object of the interface. Specific criteria will be used to guarantee an ergonomic presentation and dialogue.

- Use of formal techniques to specify the interface, the objects and their behaviour.

- Assist the designer of the interface, along the different stages of the approach, with specific computational tools.

- Validate theoretically the specification before the graphical creation (generation) of the interface.

A synthetic presentation of this approach is advanced in this paper first. Then, we focus on the formalisation of the human-machine dialogue using Petri nets. Petri nets are expressive for the event's aspect, synchronism and parallelism, as well as being pertinent criteria for the constraints of interactive graphical interfaces.

Moreover, Petri nets have already been used for human machine dialogue modelling [34] [53] or human task description [1] [24].

Here, their use is proposed for modelling the operator's behaviour and interface objects, as explained later.

3. Presentation of the proposed approach

We are especially interested in the identification of the user requirements from the HMS analysis and in the process of interface design taking these user requirements into consideration. This approach is made up of seven steps (figure 1).

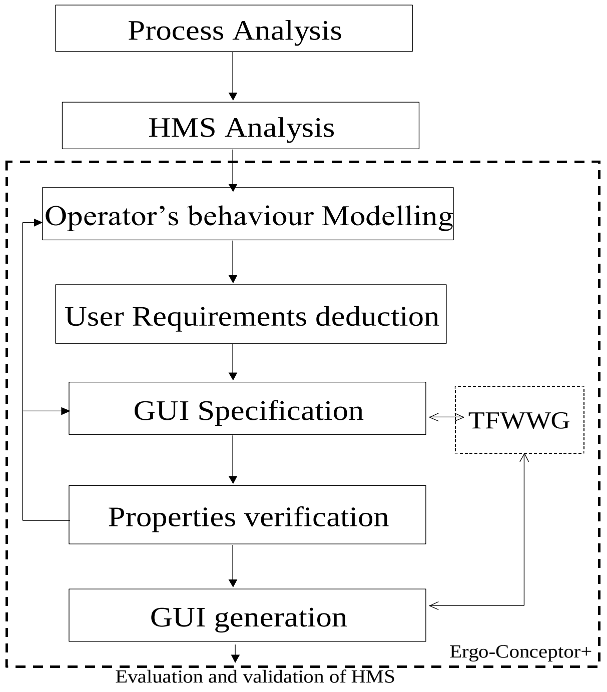


Figure 1. Methodological framework of the proposed approach

1. A first analysis of the process and its command system is necessary. This analysis provides a document containing the process data and the different technical and functional constraints.

2. The second step consists in analysing the whole HMS in terms of the process, its command system and the operator's tasks. Different methods and techniques are proposed for this. This is described in section 3.1.

3. The third step consists in modelling the operator's behaviour. It expresses the interaction of the operator with the interface. This is described in section 3.2.

4. The fourth step ensures the identification of the user requirements in terms of interface objects. This is explained in section 3.3.

5. Once the interface objects have been identified, this step consists in specifying the interface in terms of displays and graphical objects. See section 3.4.

6. We take advantage of the formal technique used for interface specification to verify the specifications generated. Much research is still required for this aspect.

7. The last step in this approach is devoted to the automatic interface generation. Preliminary results are obtained thanks to the experiment of the Ergo-Conceptor system [28] [30]. However, more research is required.

Case study

As the first application, we have chosen a simple industrial process, which consists in manufacturing metallic products. The process is controlled by an operator in the control room. It consists in (1) producing metallic sheets, then (2) coating these sheets with a chemical solution. This solution is obtained by mixing two products C and D. To prepare product C, we have to mix products A and B, which have variable temperatures and flows (TA, TB, FA and FB). The result C must have a fixed temperature and flow (TC and FC). The operator therefore has to adjust the temperature and flow of product C, by manipulating the different parameters of the system (TA, TB, FA and FB). The operator must also control the volume of product C in the tank and maintain it under a maximum level. The operator's task consists, then, in controlling the mixer functioning and intervening when a dysfunction appears in order to restore the normal functioning state of the system (see figure 2).

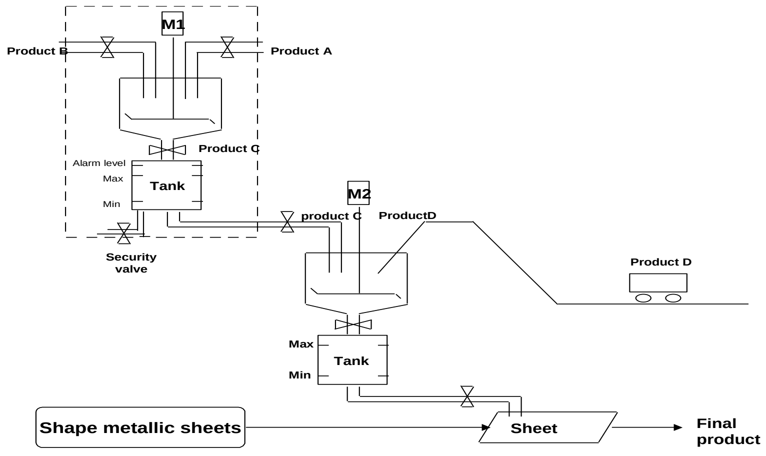


Figure 2. Process of metallic product manufacture requiring the design of control UI

We will explain below the different steps in the proposed approach and illustrate them using this pedagogic industrial example (this part extends [45]).

3.1 HMS analysis

It consists in searching the significant sub-systems and analysing their behaviour. The objective of this analysis is to identify the appropriate graphical displays. Three steps are necessary for that:

- Perform a hierarchical decomposition of the HMS. The well-known SADT (Structured Analysis and Design Technique [17]) method is proposed here. It allows us to identify the appropriate elementary sub-systems.

- Analyse the dysfunctioning of these sub-systems. For this we use an inductive method (Fault Method and Effect Analysis: FMEA [10] [42]) searching for possible faults. We then consider a deductive method (Fault Tree: FT [10] [15]) for a more precise analysis pinpointing the causes of each fault.

- Identify for each sub-system and for each functional context, the necessary operator tasks. It is assumed that the task analysis is done according to an appropriate method [40] [48]. Results of this analysis are used in our approach to define the respective action procedures. The parameters of these procedures will later allow the deduction of the user requirements.

Thus, this step consists in applying the SADT method in order to decompose the HMS and identify the appropriate elementary sub-systems. Four sub-systems can be identified in this example: (1) a first sub-system for preparing the different components (metallic sheets, products A, B and D), (2) a second sub-system for preparing the mixture C (the mixer), (3) a third sub-system is needed for preparing the chemical solution and (4) the last is for coating the sheets with the chemical solution. We choose to study the second sub-system (the mixer). We assume, then, that an analysis using the classical and complementary methods (Failure Mode and Effects Analysis method (FMEA) and Fault Tree method (FT)) has been carried out. This analysis identifies two possible faults: (1) deviation of the temperature or the flow of product C, and (2) the overflow of the storage tank.

In this way, we deduce three states: a normal functioning state, a dysfunctioning state relating to the deviation of the temperature or the flow of product C, and a second dysfunctioning state relating to the overflow of the storage tank.

We agree upon the three tasks of the human operator. We also assume that the human task analysis was carried out by human factor specialists:

- Task 1: a supervision task relating to a normal functioning.
- Task 2: a correction task to bring the temperature and/or the flow of product C into their normal values. The operator could act upon: (i) the temperatures tA, tB and, (ii) the flows fA, fB.
- Task 3: a correction task to bring the volume VC of product C in the tank to an acceptable level. At the end of section 3.2, we present the model of the operator's behaviour when faced with this dysfunctioning situation (VC>Vmax).

This preliminary analysis of the HMS aims to identify the different functioning states of the system and the possible interventions of the human operator. Petri nets are now used to model the operator's behaviour.

3.2 Operator's behaviour modelling

The aim here is to propose an approach for modelling the operator's behaviour while performing his (her) supervision and command tasks. This approach should provide a provable Petri net model in order to allow the verification of several " important" properties of the interface. The construction explained below assumes this postulation. Moreover, if it is to be easily approved by the designers, this approach has to be pedagogical and usable, especially by favouring the graphical feature of the modelling.

We have chosen Petri Nets as a formal technique for that. In fact, this technique can reliably express the aspects of concurrency and parallelism. The Petri Nets proposed here are the Interpreted Petri Nets [8]. They introduce the notion of events and conditions as well as the notion of actions. So, we associate a passing condition (Cj), a triggering event (Evj) and an possible action (Aj) to each transition (Tj) (figure 3). To model the operator's behaviour using Interpreted Petri Nets the places represent the operator's behaviour according to the system's evolution. We consider, for example, as exposed by Rassmussen [39] [40], the detection, the evaluation of the situation, the decision and the action. The evolutions between these states are modelled by the transitions.

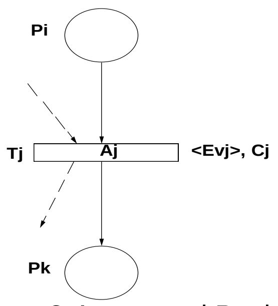


Figure 3. Interpreted Petri Net.

It should be noticed here that any change of the system state has an effect on the interface. It will imply a change in the graphical display affecting the object's parameters (colours, shapes, etc.) or the display contents (appearance or disappearance of some objects, etc.).

This modelling technique allows us to deduce the user requirements according to the functional context. An operator's task is composed of a well-organised set of elementary actions. The structure modelling the elementary action of the operator is given below (figure 4) (i.e. regulation, etc.)
The validation of the condition i (transition T1) and the presence of the event "End action" (transition T2) express that the action has been executed and has come to end.

The place P2 expresses a waiting state, while the places P1 and P3 model the state of the operator before and after the execution of the action. For example, the place P1 expresses the "mental" intention of the operator to act.

The place P3 expresses the end of the action. The actions (elementary or not) are scheduled according to typical compositions as sequential, parallel, choice, iteration, etc. We present in this paper the sequential and the parallel composition as examples.

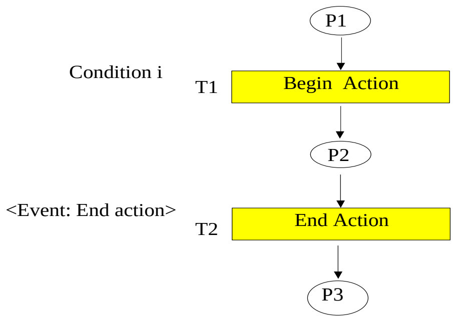


Figure 4 elementary action

Further details of these compositions will be published in future papers. The sequential composition of N actions is done by merging the output places of the Action i, and the input places of the action i+1 (figure 5).

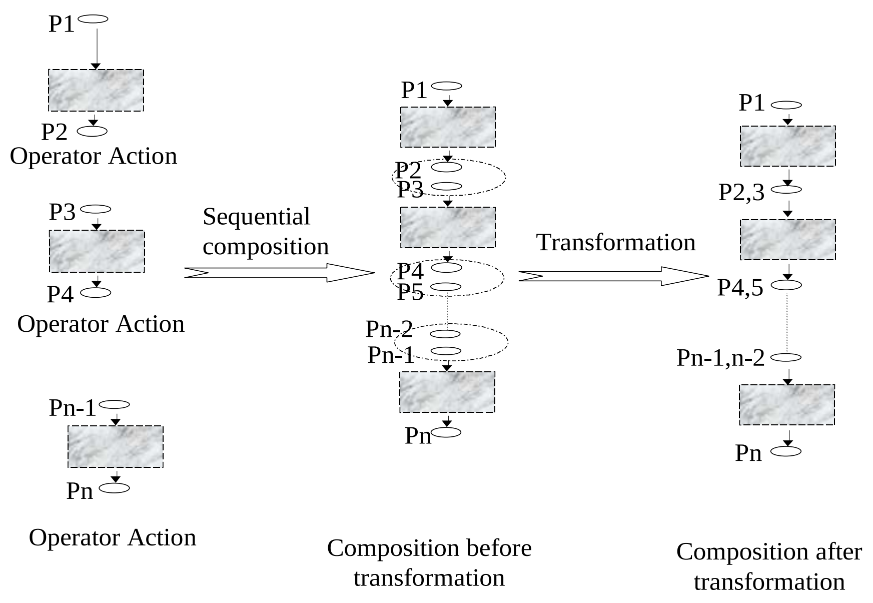


Figure 5. Sequential composition

The parallel composition of a set of actions {Ai} is done thanks to two sequential compositions:

- The first composes a structure “PAR1” with the set actions {Ai}. PAR1 assumes the simultaneous marking of all the input places of the Ai actions. (figure 6).

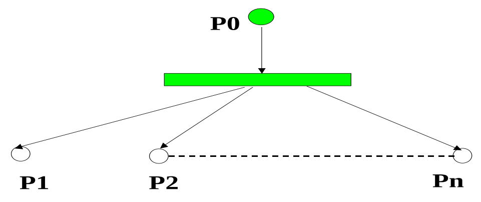


Figure 6 structure PAR1

- The second composes the actions {Ai} with the structure “PAR2”. PAR2 assumes the synchronisation at the end of all the actions (see figure 7).

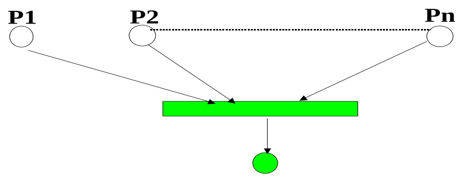


Figure 7 structure PAR2

It should be noted that the number of places Pn (in figures 6 and 7) is equal to the number of parallel actions Ai.

Thus, to ensure the parallel composition, synchronisation input and output places are needed. (figure 8).

Considering the example given above, we will illustrate the process of this modelling with the third task presented in section 3.1. This task consists in reducing the volume of product C when it exceeds the limit Vmax.

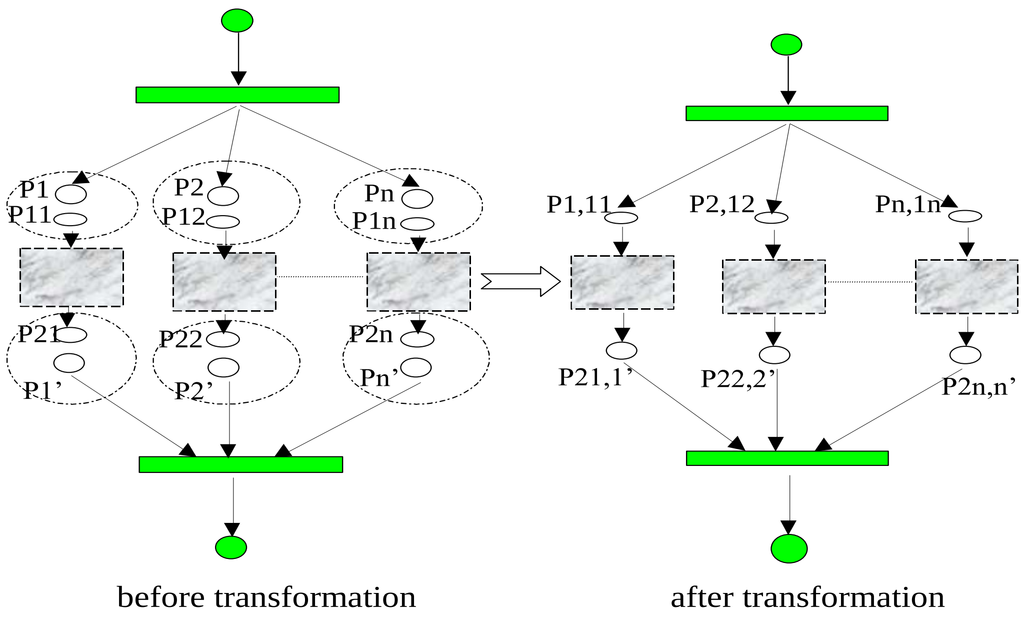


Figure 8. Parallel composition

To accomplish this task, the operator has to perform a first action in sequence with two other parallel actions (**Task correction ≡ {**A1**} Seq {**A2 **Par** A3**})**:

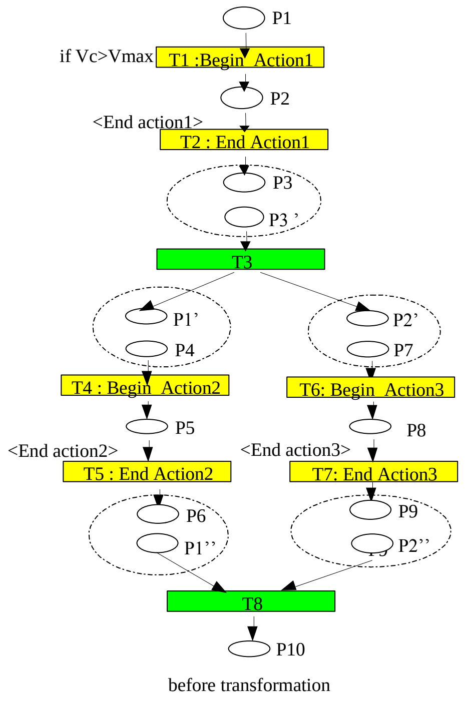
A1: empty the tank C (by opening the overflow valve associated to the tank)
A2: reduce the flow FA (by decreasing the opening rate of the associated valve)

A3: reduce the flow FB (by decreasing the opening rate of the associated valve)

The operator's behaviour when faced with each of these actions can be modelled with the elementary structure (figure 4). Thus, the model is built by composing three elementary actions (figure 9). The places P2, P5 and P8 model the waiting states of the operator's action, respectively A1, A2 and A3. The Interpreted Petri Net will be defined by the set: < P, T, E, OB, Pre, Post, μ, Precond, Action, Info-Transition > where:

- P = set of places = {P1, P2,..., Pn},
- T = set of transitions = {T1, T2,..., Tm},
- E = set of events including the event "always present" <e>,
- OB = set of graphical objects of the interface,
- Pre: P x T → N defines the weight of the bow joining a place pi of P to a transition tk of T, (figure 10),
- Post: P x T → N defines the weight of the bow joining a transition tk of T to a place pi of P,
- μ: T → E associates to each transition the appropriate triggering event,
- Precond: T → Boolean Expression defines the necessary passing condition for each transition,
- Action: T → A defines the eventual and appropriate action procedure associated to each transition
- Info-Transition: T → OB associates to each transition, the appropriate interface objects.

The latter two elements of The Interpreted Petri Net model are detailed in the following sections.



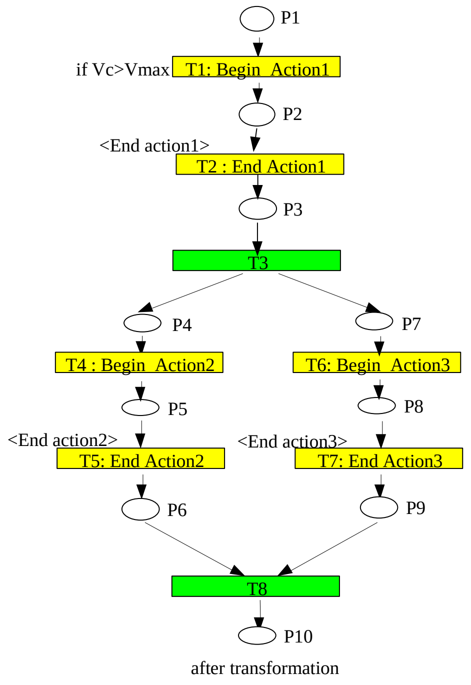
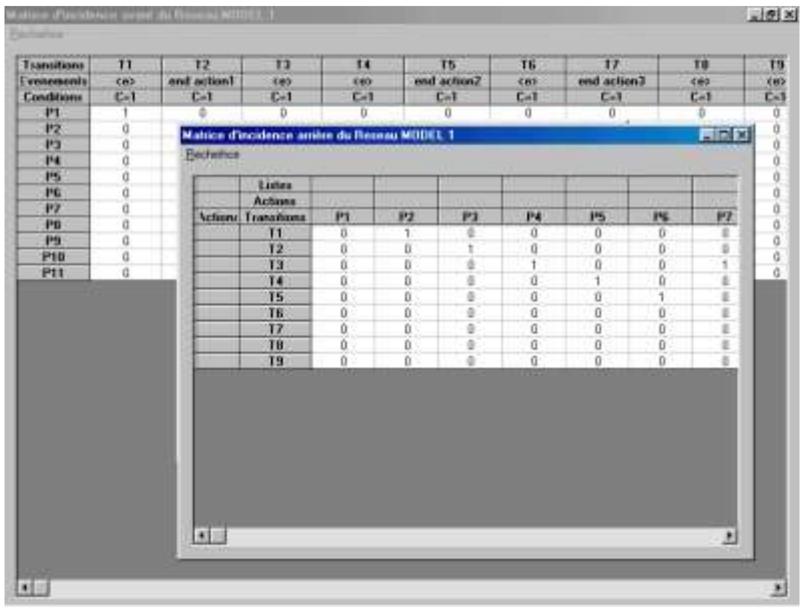


Figure 9. Operator's behaviour model



Matrice d'incidence avant du Reseau MODEL 1

Recherche

Transitions	T1	T2	T3	T4	T5	T6	T7	T8	T9
Evenements	<e>	end action1	<e>	<e>	end action2	<e>	end action3	<e>	<e>
Conditions	C=1	C=1	C=1	C=1	C=1	C=1	C=1	C=1	C=1
P1	1	0	0	0	0	0	0	0	0
P2	0								0
P3	0								0
P4	0								0
P5	0								0
P6	0								0
P7	0								0
P8	0								0
P9	0								0
P10	0								0
P11	0								0

Matrice d'incidence arrière du Reseau MODEL 1

Recherche

	Listes							
	Actions							
Actions	Transitions	P1	P2	P3	P4	P5	P6	P7
	T1	0	1	0	0	0	0	0
	T2	0	0	1	0	0	0	0
	T3	0	0	0	1	0	0	1
	T4	0	0	0	0	1	0	0
	T5	0	0	0	0	0	1	0
	T6	0	0	0	0	0	0	0
	T7	0	0	0	0	0	0	0
	T8	0	0	0	0	0	0	0
	T9	0	0	0	0	0	0	0

Figure 10. Matrices Pre and Post edited by Ergo-Conceptor+ tool

Once the operator's behaviour has been modelled, the user requirements can be deduced as shown below.

3.3 Deduction of the user requirements

Controling the process, the operator needs to know the process state instantaneously. This information will be transmitted to him by different interface objects (messages, values, graphics, alarms, etc.). Since these objects are related to the state variables of the process, we identify the appropriate set of informational variables for each state. These variables derive from the previous HMS analysis.
Moreover, in order to perform the tasks, the operator needs to intervene and command some variables in order to correct an abnormal situation. For that, the interface will present a set of control objects (or action objects) through which the operator can command the process.

The set of these command and informational variables constitute the user requirements. We explain here how to deduce the user requirements from the previous analysis. For this we use the Petri Net which models the behaviour of the operator according to functional context. We consider in the operator's behaviour model a transition «Begin Action». We associate to these transitions the adequate variables, either informational or command, which refer to the user's requirements. Thus, at the state P2, the operator disposes of the relevant user requirements to perform his action well. For instance, according to the action A1, the informational variable VC informs the operator of the current level, the command variable allows him to decrease this level by manipulating the valve CVC (figure 11).

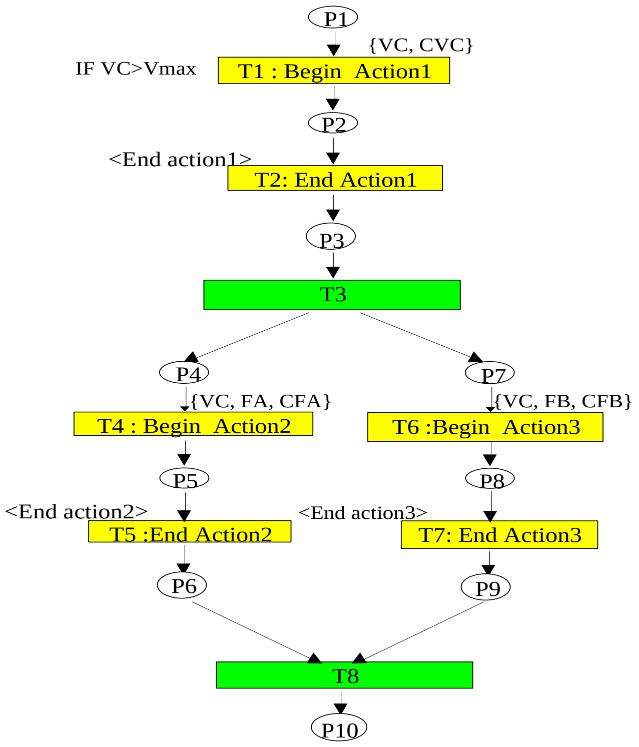


Figure 11 Operator's behaviour model within the user requirements

Once the informational and command variables have been identified, we have to deduce the necessary objects of the UI:

- to each informational variable, we associate a "graphical informational object " and
- to each command variable, we associate a " graphical command object ".

3.4 UI specification

Once graphical objects have been identified, the next step consists in specifying this UI in terms of presentation and dialogue, by using ergonomic rules (also called guidelines [47] [52]). In fact, there are many ergonomic rules available in literature. But at present, few attempts exist which aim at making an inventory of guidelines for the design and evaluation of process control interactive applications:

- Gilmore et al. [11] have strongly contributed to the first versions of the NUREG standard (devoted to nuclear applications) and have proposed a synthesis in the literature: they have written a handbook of guidelines for user-computer interface design in process control. These guidelines are organised into four categories, namely, (1) video displays, (2) control and input devices, (3) control/display integration, (4) workplace layout and environmental factors. Each guideline is presented in a structured format. Hundreds of guidelines adapted to this particular field (process control) are listed in the handbook.

- O'Hara et al. [32] [33] have contributed to the latest versions of the NUREG standard and have proposed the first version of an hypermedia system describing guidelines usable in the process control domain. Although their preliminary results are promising, these authors underline several conceptual, practical and methodological problems which are due to the specificities of the process control field.

One can also state that these guidelines are not really well-known or used in industry; nevertheless, the industrial needs and the potential projects are numerous.

For UI presentation, we propose to take advantage of the research carried out since the eighties, by some researchers [3] [20] [22] [23] [29] concerning knowledge-based approaches for automatic design and/or evaluation of UI used in process control. In particular, proposed in our previous research, Ergo-Conceptor+, a model-based tool, is able to decide on the appropriate displays to associate to each sub-system. For that, it takes into consideration, on the one hand the characteristics of each sub-system (the list of its functioning states, the user requirements associated to each state,...) and on the other hand, specific formalised guidelines stored in its knowledge bases.

The previous steps, as explained before, lead to several sub-systems where each one disposes of its variables, whether informational or command. The question here is to decide upon:

- The number of displays: according to the volume of the informational needs.
- The display types (e.g., informational, command): according to the level of abstraction of the sub-system and its role in the HMS description.
- The representation types (e.g. supervision, historical): according to the display type.

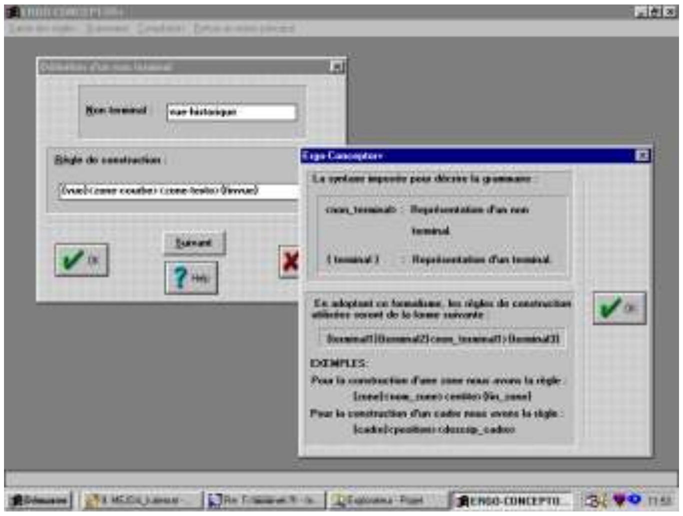
A few guidelines formalised in clausal form and then stored in its knowledge base are given below (Figure 12.a and Figure 12.b). They give examples about how it is possible to decide (1) upon the display types and representation types according to the abstraction level of the sub-system, (2) upon the choice of the graphical object representing one variable. Figure 13 shows a screen display of the Ergo-Conceptor+ system allowing the description of guidelines (ergonomic rules). The human-machine dialogue associated with the specified graphical objects is described using the Petri net model. More explanations are given below.

```
IF  level_abstraction_display (Subsys) =  “lowest”
            THEN  generate_command_display (Subsys)
            AND generate_informational_display (Subsys)
IF  level_abstraction_display (Subsys) =  “high”
            THEN generate_informational_display (Subsys)
IF number_dynamic_inform_needs (Subsys) >= MAX
            NB: (dynamic: can be displayed at the same time)
            THEN  generate_multiple (Subsys)
  NB: (generate_multiple: decompose the subsytem into N displays)
```

Figure 12.a. A few pedagogic guidelines for generating displays

```
IF type_variable (V) = “informational”
        AND type_display (subsys) = “informational”
        AND representation_type (subsys) = “historical”
        THEN  select_trend (V)
IF type_variable (V) = “informational”
        AND type_display (subsys) = “informational”
        AND representation_type (subsys) = “supervision”
        THEN  select_bargraph (V)
IF type_variable (V) = “command”
        AND type_display (subsys) = “command”
        THEN  select_Command_button (V)
```

Figure 12.b. A few pedagogic guidelines for selecting interface objects



Nom terminal : vue historique

Règle de construction :

{vue}<zone courbe> <zone texte> {finvue}

Ergo Conceptor+

La syntaxe imposée pour décrire la grammaire :

<nom_terminal> : Représentation d'un non terminal

{ terminal } : Représentation d'un terminal

En adoptant ce formalisme, les règles de construction utilisées seront de la forme suivante

{terminal1}{terminal2}<nom_terminal1>{terminal3}

EXEMPLES:

Pour la construction d'une zone nous avons la règle :

{zone}<nom_zone><entête>{fin_zone}

Pour la construction d'un cadre nous avons la règle :

{cadre}<position><descrip_cadre>

Figure 13. A screen display of Ergo-conceptor+ system

Once the different graphical displays have been identified, we have to specify the presentation and the dialogue associated to each of them. Interface objects are identified according to the user requirements deduced in step 4 of our approach. Their presentations can also be chosen according to a knowledge-based system, dealing with guidelines, in order to ensure the best ergonomic quality possible. At this level of abstraction, guidelines are more frequent and it is easier to formalise them and to automate their inference [28] [30].

It is now necessary to define the behaviour of the different graphical objects of the interface, in order to model the human-machine dialogue. It should be noted that the object’s behaviour can depend on the characteristics of the graphical object, the services it offers, the field of application, the system’s evolution, etc. Each graphical object has a control structure described by Interpreted Petri nets. We consider here the object’s states: activated, deactivated, displayed and masked.

Considering the two kinds of graphical objects identified (informational and command objects), we propose two generic control structures. The first one concerns informational objects (figure 14.a). These objects principally have two states: displayed and masked. The control structure of such an object will have two particular transitions. The actions associated to these transitions will ensure the display and the masking of the object.

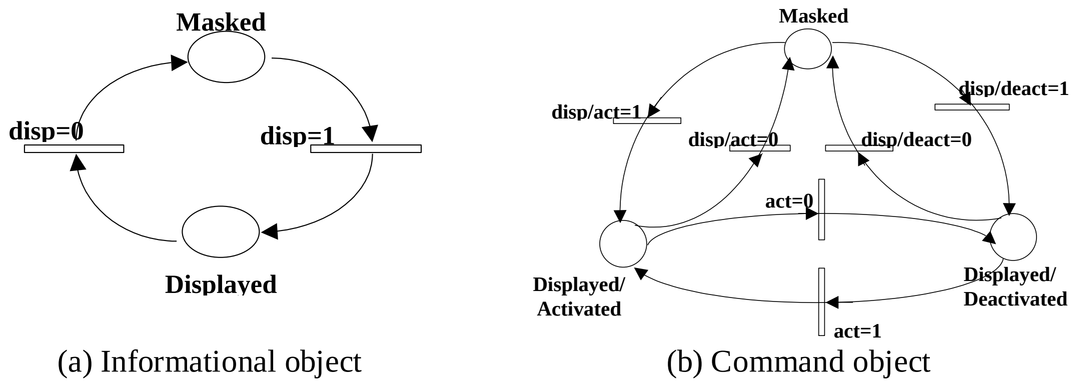


Figure 14. Control structure of UI objects

The second type of control structure concerns command objects (fig 14.b). A command object has three principle states: masked, displayed/deactivated and displayed/activated.

The global dialogue is determined by the different Petri nets proposed here: the Petri nets modelling the operator's behaviour and the different Petri nets corresponding to the control structures of different graphical objects. The communication between these PN is ensured by the use of logical variables. The states of these variables are assigned within the transitions: *Begin action* and *End action* (i.e see A={disp:=1, disp/act:=1} in figure 15). By the side of the transitions of the graphical object control structure, conditions expressed in terms of logical **F**ormula (**F**) are associated (i.e F(disp=0), F(disp/act)=1, etc. in figure 15). These formula deal with the logical variables stated in the operator's behaviour model.

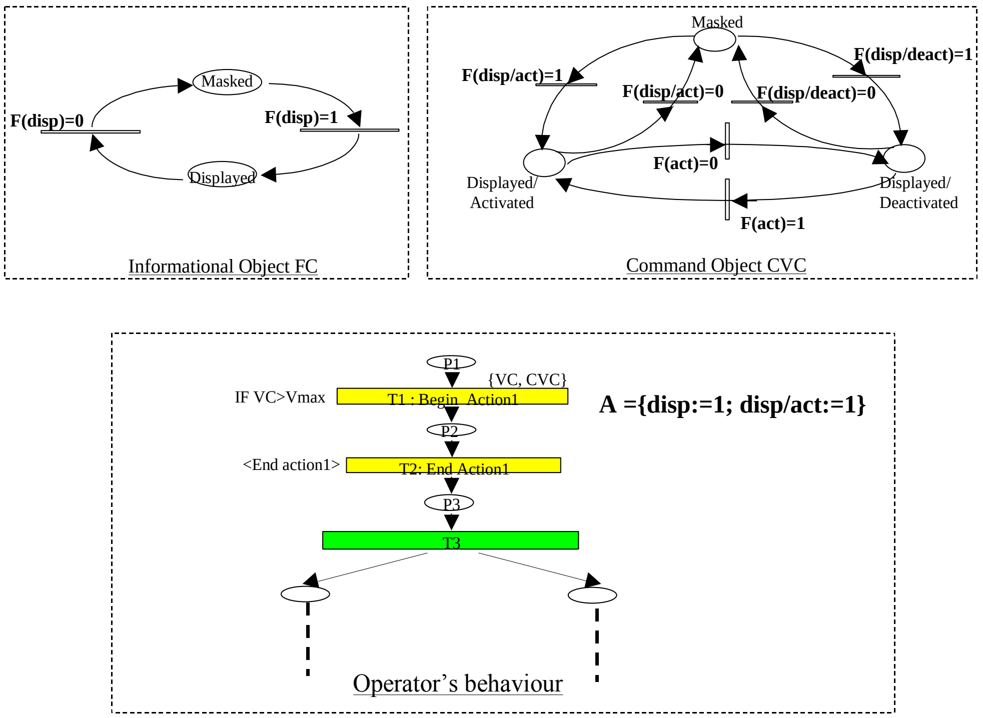


Figure 15. Communication process between different Petri Nets

Figure 15 illustrates this with the tank sub-system. It shows a Petri net modelling the operator's behaviour and the control structure of two graphical objects: An informational object, "curve", associated to the informational variable FC, a command object, "command button", associated to the command variable CVC.

3.5 Validation of the UI specifications

The validation of a model, based on PNs depends greatly on its characteristics. Indeed, usually extended Petri Nets cause difficulty in verifying several properties. The Interpreted PNs are non-autonomous. In spite of a partial lost of theoretical results (properties verification), they make it possible to express concepts of real time systems and to model concisely complex applications [4]. To validate this model, it is necessary to add several constraints. We consider, mainly, the events coming from the human operator, always present; his action must come to an end in any case. Thus, the external events will never cause a deadlock of the PN model.
We can, therefore, move towards to an ordinary PN by removing all the external events. In this way, the properties verification becomes achievable. Furthermore, as the construction of the global model is based on the merging technique of the places, the composition conserves the properties verified on the elementary structures.

In other words, by composing validated elementary structures, the global model obtained is itself valid. It should be noticed that the elementary structure verifies several properties. It is bounded, alive, without conflict, without deadlock, etc.

Further research needs to be carried out concerning this validation step.

3.6 UI generation

The last step in this approach is devoted to automatic UI generation. First results were obtained (figure 16.a and 16.b) during the design of the first version of Ergo-Conceptor [28] [30]. These results show the feasibility of this approach, but consistency is still required. The approach followed is based on a knowledge-based approach. It is presented in detail in [30].

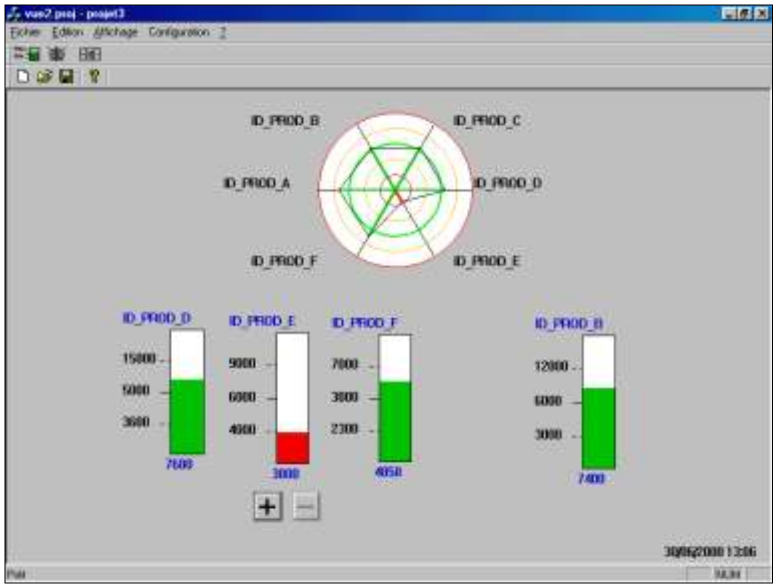


Figure 16.a. A graphical display generated by Ergo-Conceptor+ stating a dysfunctioning of the system (on the variable ID-PROD-E)

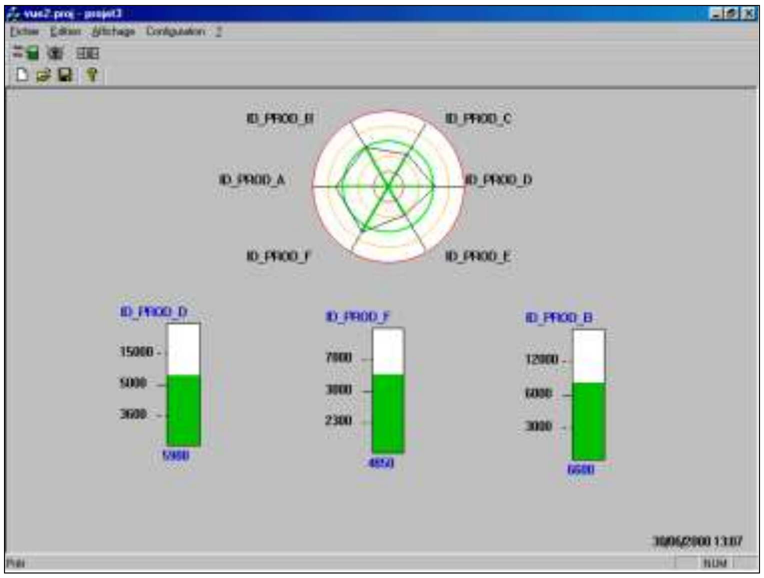


Figure 16.b. A graphical display generated by Ergo-Conceptor+ stating a normal functioning of the system.

4. Conclusion

We have concentrated in this paper on one of the more difficult steps in any global methodology for UI design: the modelling of human-machine dialogue. We have explained, through an industrial case study, the principle of this modelling using interpreted Petri nets.

We are concentrating now, particularly, on the steps of graphical specification and automatic generation of UI. We study for that the contribution of original approaches such as multi-agent systems, distributed artificial intelligence, etc.

Acknowledgements

The authors would like to thank Nawel KHELIL for her contribution in this research..

References

[1] M. Abed, Contribution à la modélisation de la tâche par outils de spécification exploitant les mouvements oculaires: application à la conception et à l'évaluation des interfaces homme-machine. Ph.D. Dissertation, University of Valenciennes, France, 1990.

[2] Y. Ait-Ameur, P. Girard and F. Jambon, A uniform approach for specification and design of interactive systems: the method B, *Proceedings DSV-IS'98*, Coesner's House, Abingdon, UK, June 3-5 1998, pp. 333-352.

[3] J.L. Alty, P. Elzer, O. Holst, G. Johannsen, S. Savory, Literature and user survey of issues to man-machine interfaces for supervision and control systems. *Report of Esprit project P600*, Copenhagen: C.R.I., 1985.

[4] G.W. Brams, Réseaux de Petri: Théorie et Pratique, Tome 2: modélisation et application, Editions Masson, Paris, 1983.

[5] S.K. Card, T.P. Moran and A. Newell, *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum, 1983.

[6] J. Coutaz, Interfaces Homme-Ordinateur: Conception et réalisation, Editions Dunod Informatique, Bordas, 1990.

[7] B. D'Ausbourg, G. Durrieu and P. Rocher, Deriving a formal model of interactive system from its UIL description in order to verify and to test its behaviour, *in proceedings of DSV-IS'96*, springer verlag, pp. 105-122.

[8] R. David and H. ALLA, *Du GRAFCET aux Réseaux de Petri*, Editions HERMES, Paris, 1992.

[9] P. De Rosis, Formal Description and Evaluation of User Adapted Interfaces, Int. Journal for Human Computer Studies, vol 49, 1998 pp 95-120

[10] E. Fadier, Fiabilité Humaine: Méthodes d'analyse et domaine d'application. In: Leplat, J., de Terssac, G. (Eds.): Les facteurs humains de la fiabilité dans les systèmes complexes. Editions Octarès, Marseille (1990) 47-80

[11] W.E. Gilmore, D.I. Gertman and H.S. Blackman, *User Computer Interface in process Control: A Human factor* Engineering Handbook, Academic Press, 1989.

[12] D. Grammenos, D. Akoumianakis, C. Stephanidi, Integrated support for working with guidelines: the guideline management system, Interacting with Computers 12, 2 (1999) 281–311

[13] D. Harel, Statecharts: a visual formalism for complex systems. In *Science of Computer Programming*, volume 8, 1987.

[14] M. Harrison and M.Thimbelbey, *In formal methods in human computer interaction*, Harrison & Thimbelby (Eds) 90, Cambridge university Press, 1990.

[15] D.F. Hassl, Advanced concepts in fault tree analysis. *System Safety Symposium*, Seattle, 8-9 Juin 1965.

[16] D. Heller, P.Ferguson and D.Brennan, Motif Programming manual, 2nd edition, February 1994.

[17] I.G.L. Technology, *SADT, un langage pour communiquer*. Eyrolles, Paris, 1989.

[18] R.J.K. Jacob, A state transition diagram language for visual programming, *IEEE Computer*, vol. 18 (8), August 1985, pp. 51-59.

[19] R. Jeffries, The role of task analysis in the design of software, In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 347-359, 1997.

[20] G. Johannsen, Conceptual design of multi-human machine interfaces. *Proceedings 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems*, MIT, Cambridge, USA, June 27-29, 1995.

[21] C. Kolski, Interfaces homme-machine, application aux systèmes industriels complexes. Editions HERMES, Paris, 1997.

[22] C. Kolski and P. Millot, A rule-based approach for the ergonomic evaluation of man-machine graphic interface. *International Journal of Man-Machine Studies*, 35, pp. 657-674, 1991.

[23] C. Kolski and F. Moussa, Two examples of tools for working with guidelines for process control field: Synop and ERGO-CONCEPTOR, *Third Annual Meeting of the International Special Interest Group on "Tools for working with guidelines"*, J. Vanderdonckt (Ed.), Namur, Belgium, 4 June 1996.

[24] A. Mahfoudhi, TOOD: Une méthodologie de description orientée objet des tâches utilisateur pour la spécification et la conception des IHM: Application au contrôle du trafic aérien, Ph.D. Dissertation, University of Valenciennes, France, 1997.

[25] A. Marcus, Graphical user interfaces. In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 423-440, 1997.

[26] J.P. Meinadier, L'interface Utilisateur pour une informatique plus conviviale, DUNOD Informatique, Paris, 1991.

[27] J.F. Monin, *Comprendre les méthodes formelles: panorama et outils logiques*, Edition Masson, Paris, 1996.

[28] F. Moussa, *Contribution à la conception ergonomique des interfaces de supervision dans les procédés industriels, application au système ERGO-CONCEPTOR*. PhD Dissertation, University of Valenciennes, 1992.

[29] F. Moussa, C. Kolski and P. Millot, Artificial intelligence approach for the creation and the ergonomic design of man-machine interfaces in control room. Ninth European Annual Conference on "Human decision making and manual control", Varese, Italy, September 10-12, 1990.

[30] F. Moussa, C. Kolski, M. Riahi, A Model Based Approach to Semi-Automated User Interface Generation for Process Control Interactive Applications. Interacting with Computers 12, 3 (2000) 245–279.

[31] F. Moussa, M. Riahi, M. Moalla and C. Kolski, C., A petri net method for specification and automatic generation of user interface. In « Human-Computer Interaction: Ergonomics and user interfaces, volume 1, Proceedings HCI'99, 8th International Conference on Human-Computer Interaction, Munich, Germany, August », H.J. Bullinger, J. Ziegler (Eds.), Lawrence Erlbaum Associates, 1999, pp. 988-992.

[32] J. O'Hara, W. Brown, W. Stubler, J. Wachtel and J. Persensky, Human-Machine Interface Design Review Guideline (NUREG-0700, rev. 1). U.S. Nuclear Regulatory Commission, Washington, 1996.

[33] J. O'Hara and W. Brown, Software Tool for the Use of Human Factors Engineering Guidelines to Conduct Control Room Evaluations. In: Büllinger, H.J., Ziegler, J. (Eds.): Human-Computer Interaction: communication, cooperation and application design. Lawrence Erlbaum Associates, London, 973–977, 1999.

[34] P. Palanque, Spécifications formelles et systèmes interactifs: vers des systèmes fiables et utilisables. *Habilitation à diriger des recherches,* University of Toulouse I, 1997.

[35] P. Palanque and F. Paterno (Eds.), Formal Methods in Human-Computer Interaction, Springer Verlag, 1997.

[36] W.J. Pardi, XML en action, Microsoft Press, 1999.

[37] D.L. Parnas, On the use of transition diagrams in the design of user interface for an interactive computer system, *24 th ACM Conference,* 1969, pp. 379-385.

[38] F. Paterno and G. Faconti, On the use of Lotos to describe graphical interaction, *In proceedings of people and computer VII, HCI'92 conference*, cambridge university press, 1992, pp. 155-174.

[39] J. Rasmussen, Intelligent Decision Support in Process Environments. A framework for cognitive Task Analysis in System Design In: Hollnagel, E., Mancini, G., Woods, D.D. (Eds.). NATO ASI series. Vol. F21. Springer-Verlag, Berlin (1986).

[40] J. Rasmussen, Information processing and human-machine interaction, an approach to cognitive engineering. *Elsevier Science Publishing, 1986.*

[41] J. Reason, *Human Error.* Cambridge: Cambridge University Press, 1990.

[42] J.L. Recht, *Failure mode and effect*. National Safety Council, 1966.

[43] P. Reed, K. Holdaway, S. Isensee, E. Buie, J. Fox, J. Williams and A. Lund, User interface guidelines and standards: progress, issues, and prospects, Interacting With Computers, 12, 2 (1999) 119–142.

[44] M. Riahi, F. Moussa, M. Moalla and C. Kolski, Vers une spécification formelle des interfaces homme-machine basée sur l'utilisation des réseaux de Petri. In M.F. Barthet (Ed.), Actes du 6ème Colloque ERGO IA'98 Ergonomie et Informatique Avancée. (pp. 196-205). Bayonne: ESTIA/ILS, 1998.

[45] M. Riahi, F. Moussa, C. Kolski and M. Moalla, Use of interpreted petri nets for human-machine dialogue specification in process control. Proceedings ACIDCA'2000 International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications, 22-24 March, Monastir, Tunisia.

[46] T.B. Sheridan, Task analysis, task allocation and supervisory control. In Handbook of Human-Computer Interaction, Helander M.G., Landauer T.K., Prabhu P. (Eds), North Holland, Amsterdam, pp. 87-105, 1997.

[47] S.L. Smith and J.N. Mosier, Guidelines for designing user interface software. Report EDS-TR-86-278, The MITRE Corporation, Bedford, MA, 1986.

[48] R.B. Stammers, M.S. Carey and J.A. Astley, Task analysis. In *Evaluation of human work. A Practical Ergonomics Methodology*, Wilson J. R. and Corlett E. N. (Eds.), Taylor & Francis, 1990.

[49] D. Tabary and M. Abed, TOOD: an object-oriented methodology for describing user task in interface design and specification - An application to air traffic control. La Lettre de l'Intelligence Artificielle, vol 134-135-136, pp 107-114, 1998.

[50] K.J. Turner, Using formal description techniques: An introduction to Estelle, Lotos and SDL, edition John, 1993.

[51] J. Vanderdonck, Guide ergonomique des interfaces homme-machine. Presses Universitaires de Namur, Namur, Belgium, 1994.

[52] J. Vanderdonckt, Development Milestones toward a Tool For Working With Guidelines. Interacting with Computers 12, 2 (1999) 81–118.

[53] R. Williem and V. Biljon, Extending Petri Nets for specifying Man-Machine dialogues, International Journal of Man-Machine Studies, vol. 28, 1988, pp. 437-45