A PNML Extension for the HCI Design

Faouzi Moussa *faouzimoussa@gmail.com*
National School of computer sciences
University of Manouba
Manouba, 2035, Tunisia

Ines Riahi *ines.riahi@yahoo.fr*
National School of computer sciences
University of Manouba
Manouba, 2035, Tunisia

Meriem Riahi *meriem.riahi@insat.rnu.tn*
National School of computer sciences
University of Manouba
Manouba, 2035, Tunisia

Abstract

Our research aims to propose a global approach for specification, design and verification of context awareness Human Computer Interface (HCI). This is a Model Based Design approach (MBD). This methodology describes the ubiquitous environment by ontologies. OWL is the standard used for this purpose. The specification and modeling of Human-Computer Interaction are based on Petri nets (PN). This raises the question of representation of Petri nets with XML. We use for this purpose, the standard of modeling PNML. In this paper, we propose an extension of this standard for specification, generation and verification of HCI. This extension is a methodological approach for the construction of PNML with Petri nets. The design principle uses the concept of composition of elementary structures of Petri nets as PNML Modular. The objective is to obtain a valid interface through verification of properties of elementary Petri nets represented with PNML.

Keywords: Human-Computer Interaction, Formal Specification, Ubiquitous Environment, Ontology, OWL, Petri Nets, PNML, XML.

1. INTRODUCTION

Last years, we attended the emergence of ubiquitous computing following the evolution of new mobile technologies [1], [2], [3]. This led to the increasing use of mobile devices in applications.
The adaptation of the HCI to these new supports became necessary.
Indeed, the adaptation of the human-computer interfaces to an ubiquitous environment consists in producing interfaces which can adapt themselves to a various types of mobile terminals in a dynamic way while respecting these ergonomic properties [4], [5], [6].
Up to now, the majority of the works focused essentially on the technological aspects of the mobile terminals or on the problems of evaluation of the usability of the mobile devices forgetting an essential point which consists of the modeling of the mobile HCI [7], [8]. Our research fits within this context. They aim at the elaboration of an approach of specification, design and verification of HCI.
We use a Model Based Approach. Indeed, last years, we see a craze around MBD (Model Based Design/Development) approaches, ([9]), This trend has accelerated with the proposal of the approach MDA (Model Driven Architecture) by the OMG and the Model-Driven Engineering (MDE) [10], [11], [12]. This class of approaches necessarily involves the use of models (process, task, interaction, etc..) in the design process. Our approach follows this trend. It reserves a large part to the modeling.
This Approach is made up of five steps [13], [14], [15]:

- The first step consists in analyzing the whole Human-Computer System in terms of the system (in different contexts and environments), the interaction and the user's tasks. The modeling of the System's behavior becomes possible. It expresses the interaction of the User with the Graphical Interface. This analysis is carried out using Petri Nets modeled in XML.
- The second step is the achievement of the deduction of the user requirements. In fact, the ubiquitous environment analysis and its modeling provide the set of user requirements in accordance with each functional context.
- The third step ensures the identification of the interface objects according to the user requirements. Once the interface objects are identified, this step consists in specifying the interface in terms of displays, graphical objects and dialogue.
- In the next step, we take advantage of the formal technique used for interface specification to verify critical properties of the generated specifications. Indeed, formal techniques are specially recommended since they allow the designers to proceed to the validation (even mathematical) of the UI (User Interface) before going on to their actual creation and implementation.
- The last step in this approach is devoted to the automatic interface generation.

In the first step of our approach (the analysis of human-machine system in a ubiquitous environment) we study the environment in order to identify the functional characteristics in different contexts. Then, we identify and we analyze for each context, the user's tasks.
This analysis allows defining the user's actions according to these different contexts [16], [17], [2], [18].
The parameters of these actions constitute, therefore, the user requirements. Thus, this analysis will provide a document containing the different variables necessary for the HCI modeling.
Afterwards, we proceed to the modeling of the human-computer interaction in terms of user actions. This modeling takes into account the different contexts and the evolution of the environment. It is done with a formal technique. This allows the validating of some important properties of the interface. Petri Nets are proposed for this purpose. However, this technique raises the problem of how to integrate the Petri nets with the XML documents obtained by the description of the context (OWL)?
This question was raised and treated, partially, during our previous works. Indeed, Petri Net Markup Language (PNML) is used for this purpose. PNML is used, here, as being a translator of Petri Nets in XML and not as being an exchange format between these last ones. Thus, the phases of analysis and modeling of a HCS leads to an XML document containing all the variables necessary for the modeling as well as for the graphic generation of the interface.
The mechanism of information exchange between the XML schemas describing (i) the analysis phase of the context and (ii) the modeling phase of the HMS has been studied and an algorithm was proposed for this purpose.
In this paper, we propose, first, the principle of modeling human-computer interaction with the PN and we raise the problem of integrating the PN with the use of ontologies.
We present then a methodology of modeling and validation of human-computer interaction based on a composition of elementary structures of PN.
This methodology forms the basis of a proposed extension of the standard PNML and represents our essential contribution of this article.

2. MODELING OF THE HUMAN-COMPUTER INTERACTION

2.1 Petri Nets for HCI

We looked, through a literature review, to demonstrate the ability of PN to respond to important and applicable criteria for the specification of HCI.
The use of Petri nets for modeling aims at preparing the ground for formal verification and prior validation of interfaces, which saves considerable time in the development cycle of HCI.
Indeed, Petri nets have a formal definition, they offer a great ability to express such aspects as timing, concurrency, etc., they are enforceable and have many techniques for an automatic verification of properties (boundedness, liveness , resettability, etc..).

They offer, in addition, an unconstrained graphic representation. So many criteria adapted to the constraints of specification and of automatic generation of the HCI.
In the literature, the PN are constantly expanding and represent a suitable tool for modeling human-computer interaction.
At the beginning, they were used only for the description of tasks to be computerized. But afterward, and especially with the emergence of the high-level Petri nets, we begin to take advantage of their power to model the Human-Computer dialogue. Willem and Biljon were the first to use the Petri net for the modeling of the Human-Computer dialogue [19]. Palanque uses the PN in the formalism ICO [20], [21]. Tabary proposes it in TOOD [22] (following upon the works of [23]). Finally, De Rosis used Petri nets in XDM [24].
We propose, below, a list of criteria that have led to the adoption of Petri nets in our work (more details are available in [14], [25] :

- *The power of the tool in terms of verifiability of the properties*: the verification on model allows assuring, in a sure way, the properties. This method requires that the properties are defined beforehand formally. Petri nets allow to make sure, by investigating the graph of the markings, that the user indeed has access to all the commands of the interface [26] or that he can finish a begun action;
- *The degree of the taking into consideration of the parallelism aspect:* the competition in the modeling of the user tasks is a very important criterion. We can model the competition by using the true parallelism or the interlacing. Petri nets, as the temporal logic XTL [27] or the notation UAN [28] are examples which perform this property;
- *The degree of the taking into consideration of the synchronism aspect*: to implement the synchronization of the actions and the processes is indispensable in the modeling of the Human-Computer dialogue. Petri Nets, Lotos and Lustre assures this kind of criterion [29], [30], [31], [32];
- *The sequencing of the actions and the temporal constraints:* this criterion include five sub-criteria [33] allowing to specify the relations between the actions: the sequence between two tasks (A then B); the choice between two tasks (A or B); the composition of two tasks in any order (A and B); the iteration (In a certain number of times); the expression of deadlines between the tasks;
- *The generative capacities:* according to the methodological objectives that we become attached, an approach of conception can lead to unusable informal specifications in an automated way, as she can succeed on the generation automatic or semiautomatic of the HCI from the obtained specifications. Although we often simply use a notation if it is sufficiently expressive for our needs, the generative capacities remain the specificity of the formalisms. For example, the Petri nets, whose marking graph provides the current state of the system, it's possible evolution, as well as the previous steps, allow to generate of the contextual help by explaining to the user how it reached the current state, how he can continue his task and even give him the optimal path of realization comprising the fewest possible steps [20].

In our works, we use the Interpreted Petri nets for the modeling of the human-computer interaction [34]. As mentioned before, we work on context-sensitive HCI. We commonly use ontologies in this area. An ontology is a formal system whose purpose is to represent the knowledge of a specific domain by means of concepts defined and organized ones compared to the others [35], [36], [37].
The specification of the ontologies is often based on semantic Web languages. OWL is the most frequently used language. A document OWL is an XML dialect which consists of sets of classes, hierarchies of classes, properties, constraints on these elements and types of relations allowed on these entities [38]. Thus, we need to deal with context-sensitive Petri Nets. In that sense, a literature of XML Petri nets is now presented.

2.2 Petri Nets With XML

In last years a new variant of high-level Petri nets was born: the XML nets. This was because of the fast growth of Internet, mobile medias, languages for modeling [38]. The XML nets insure the exchange of the electronic documents between heterogeneous platforms.

In addition to the permanent birth of new variants of the Petri nets, the standardization of this language was perceived as an opportunity to obtain a better organization of the work in the community of the Petri nets. Matthias Jûngel [39] presented the concepts and the terminology of PNML : Petri Net Markup Language. PNML provided a starting point for developing a standard exchange format for the Petri nets. Several improvements of a standard are listed below:

- Allow the researchers and the engineers to use the same terminology;
- Develop future extensions on a stable common base;
- Provide a reference implementation which will facilitate the data exchange between the various tools of Petri nets thanks to a common format.

We present, first, XML nets, a variant of high-level Petri Nets, then, Petri Net Markup Language (PNML), an exchange XML format between Petri Nets.

2.2.1 XML Nets

XML Nets represent a new variant of high-level Petri Nets. Its main strength lies in its power to use formal semantics with graphic nature for the modeling of the exchange of data based on XML. XML Nets identify and decomposes the process into a set of XML fragments and assign them to the adequate organizational unit [38]. Besides, the XML nets are adapted to the Web Services Composition (WSC) [40]. They also present, advantages for the management of supply chains [41] and to support all the activities in the inter-organizational management of the business processes [42].

However, the XML Nets have no means of verification of the Petri nets. In addition, the representation of complex systems leads to complex XML schemas are difficult to exploit. Therefore, this modeling technique doesn't answer our objectives. That's why we are interested in the standard PNML.

2.2.2 PNML

During the last decade, several theories and applications of exchange between Petri nets based on XML have emerged. Among those we can distinguish the PNML standard (Petri Net Markup Language, ISO / IEC 15909). The design of the PNML was governed by the following principles [43]:

- Readability: The format must be human readable and editable by any text editor,
- Universality: The format should be able to represent any kind of Petri nets with its possible extensions and special features;
- Mutuality: The format should allow us to extract as much information as possible from a Petri net. Therefore, the format must extract the common principles and the common notations of Petri nets [44].

The use of XML guarantees the readability of the format. Universality can be guaranteed by labeling net objects and the net itself. The legal labels, their possible values, and the possible combination of values are defined by a Petri Net Type Definition (PNTD). Mutuality can be guaranteed by conventions, which are a set of standardized labels [45].

The main idea of PNML is that any kind of Petri net can be considered to be a labeled graph. In particular, all information that is specific to a particular kind of Petri net can be captured in labels. His basic structure of a document PNML is defined in PNML Core Model [46]. This model imposes no limitation on labels so he can represent every types of Petri net.

2.2.2.1 PNML Core Model

A document which answers the requirements of PNML Core Model is called a document of Petri net (PetriNetDoc). It contains one or several PN. Every PN have a unique identifier and a type. A Petri net consists of one or more pages which in turn are composed of objects.

Objects: every Petri net consists of objects which represent the graphic structure of the network. Every object possesses a unique identifier which can be used to make single reference, a transition or an arc.

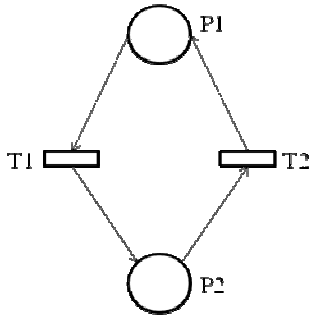
Labels: they were conceived to attribute a sense to every object. A label represents the name of a node, the initial marking of a place, the state of a transition or an arc. There are two sorts of labels: annotations and attributes. An annotation is posted as a text next to the object contrary to the attribute which has an effect on the shape or the color of the corresponding object.

The graphical information: can be associated with every object and with every annotation. They concern the size, the color, the font and the position of a node or an arc.

The specific information tools: for certain tools, it may be necessary to store certain internal information relative to the used tool. To do this, every object and every label can be equipped with the tool-specific information. This last one is not specified by PNML, it is just enough to put the name of the tool as well as its version.

Pages and reference node: a page can contain several pages or several objects. PNML requires that an arc has to connect nodes on the same page. If we need to connect two nodes present on two pages then, one of them migrates to the other page. We call it, then, reference node. Reference places and reference transitions have to refer, respectively, to a reference place / place and to a reference transition / transition [46].

To be interpreted without ambiguity and to insure the property of compatibility, every definition of a type of PN has to have one formal semantics which has to be known by every tool which it uses [47]. The description of semantics and functions of the PN and their types is not defined by this document. The notions of PNML supply a starting point to this description. If we need to import a PN whose semantics are unknown, which means that our tool does not take him in charge, a manual entry can be performed or the tool can extract a maximum of information by guessing their meanings from the names of their labels what can involve a bad interpretation. To remedy this problem, PNML offers interface definition functions. Every element fixes the syntax as well as its label's semantics. All this information is stored in the conventions document. By consulting this last one, a tool can then know the labels meaning and identify so the unknown PN to convert it in a standard PN [44]. Figure 1 illustrates an example of Petri net with its code PNML.




```
<pnml xmlns="http://www.informatik.hu-
berlin.de/top/pnml/ptNetb">
<net id="exemple1" >
   <place id="p1">
      <name><text>p1</text></name>
   </place>
   <place id="p2">
      <name> <text>p2</text></name>
   </place>
   <transition id="t1">
      <name><text>t1</text></name>
   </transition>
   <transition id="t2">
      <name><text>t2</text></name>
   </transition>
   <arc id="a1" source="p1" target="t1"></arc>
   <arc id="a2" source="t1" target="p2"></arc>
   <arc id="a1" source="p2" target="t2"> </arc>
   <arc id="a1" source="t2" target="p1"> </arc>
</net>
</pnml>
```

FIGURE 1: Example of PNML file

2.2.2.2 PNML Modular

PNML Modular is considered as an extension of PNML [45]. It was implemented to address a major problem: the size of a XML document. Indeed, the amount of information of the real world systems is considerable and cannot therefore be represented on the same page. Thus, it would be wise to divide this information into a set of modules. The use of this concept is practical because once defined, a module, represented in Figure 2, can be repeatedly used just like a call of a function within a program. A complex system can thus be constructed recursively from instances of different modules.

```
<module name="M1">
    <interface>
        <importPlace id="p1"/>
        <exportPlace id="p2" ref="y"/>
    </interface>
    <referencePlace id="x" ref="p1"/>
    <transition id="t1"/>
    <transition id="t2"/>
    <place id="y"/>
    <arc source="x" target="t1"/>
    <arc source="t1" target="y"/>
    <arc source="y" target="t2"/>
    <arc source="t2" target="x"/>
</module>
```

FIGURE 2: Example of a module PNML

PNML Modular introduced, besides the concepts of PNML Core Model, other concepts as symbols, identifiers, modules, import and export and finally modules instances [48]. Whatever the nature of PNML is, this standard is based on the XML markup language. Petri nets, objects and labels are represented as XML elements. An XML element can have several attributes to express more information.

In regard of our objectives, it seems that the XML Nets cannot suit us in any way. Thus, we opt for the PNML standard. Indeed, it provides a framework open and rich enough to meet the modeling constraints of the human machine system with Petri nets. However, for complex applications, the global Petri net of the human-computer interaction obtained with PNML reaches an important size making it difficult to exploit and to validate. PNML Modular proposes, in this sense, to use the modules. This allows reducing the size of the PNML scheme. But the call to a module (like a computer function) does not represent an effective methodological contribution to guarantee the good construction of a model of the Human-Computer system. Its contribution to give concise and confirm plans remains quite limited.

Besides, PNML Modular cannot guarantee the exactness of the syntactic construction of Petri net. For example, the code PNML of Figure 3 describes an incorrect definition of a Petri net where two places are connected by an arc. In that case, PNML is not capable of verifying that the source attributes and arc targets are of type nodes and that these nodes are of different types (place or transition) [49]. A methodological shortcoming that we strive to overcome through our work.

```
<pnml xmlns="http://www.informatik.hu-berlin.de/top/pnml/ptNetb">
<net id="example1" >
 <name>
        <text>incorrect representation of Petri net</text>
</name>
<place id="p1">
    <name><text>p1</text> </name>
</place>
<place id="p2">
    <name><text>p2</text> </name>
</place>
<arc id="a1" source="p1" target="p2">
    <inscription><text>1</text> </inscription>
</arc>
</net>
</pnml>
```

FIGURE 3: Incorrect definition of Petri Net

In this regard, our philosophy of modeling human-computer interaction is based on compositions of elementary structures of Petri nets simple, verifiable and each fulfilling a particular role. This proposition of extension aims at extending, in a way, the principle of PNML Modular. It is an

extension (i) on the syntactic plan by offering Petri nets that are much more structured and easy to use and (ii) on the semantic plan, by offering valid Petri nets in terms of verification of strong properties of the HCI.

2.3 Petri Nets for Modeling Human-Computer Interaction

As we mentioned above, our approach to model human-computer interaction provides a verifiable Petri Net model on which we can validate important properties of the interface. The model design principles proposed in our work guarantee this postulate. Moreover, this approach is pedagogical and easy to use giving it the ability to be easily approved and adopted by the designers. This is ensured by promoting simplicity and the graphic aspect of the modeling [34], [50].

We, therefore, chose the formal technique of Petri nets for modeling the user's behavior. Indeed, it is considered in the literature that this technique is efficient for the expression of synchronization, parallelism and competition. The Petri net used here is IPN (Interpreted Petri nets) [51]. This type of networks introduces the notions of event, condition and the notion of action. Indeed, a passing condition (Cj), a trigger event (Evj) and a potential action (Aj) are associated with each transition Tj of an IPN (Figure 4).

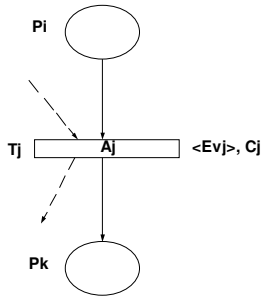


FIGURE 4: Interpreted Petri net

To model the human-computer interaction using Petri nets, we agreed to use the places to represent the state of the system depending on the environment evolution.

We refer to Rasmussen and Norman [52], [53] in the human decision theory of the operator when solving a problem. This theory states that when solving a problem, the human operator goes through four phases: (i) detection, (ii) evaluation, (iii) decision and (vi) action. We model, in our work, the evolution between these states with the PN transitions.

We consider that a user task is composed of an organized set of elementary actions. The structure modeling an elementary action of the user is presented in Figure 5. The validation of the condition i (transition T1) models the fact that the user will start the execution of the action relative to that condition. Later, the happening of the event "end action" (transition T2) expresses the fact that the user action was performed and ended. The place P2 expresses a waiting state for the end of the action's execution, while the places P1 and P3 model the state of the user before and after the execution of his action. For example, P1 models the user's mental intention in order to act. The place P3 expresses his state at the end of the action's execution.

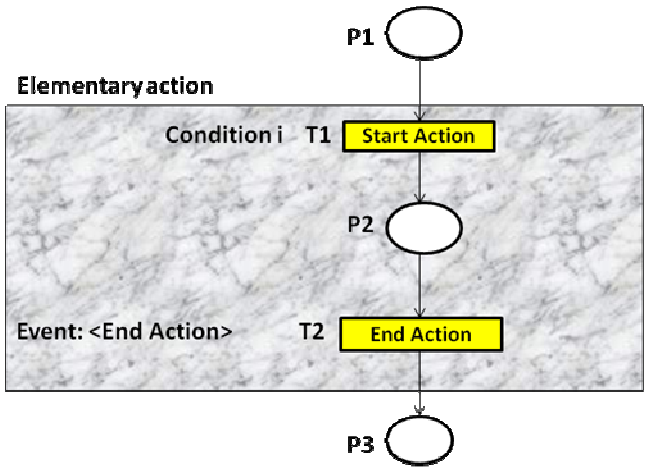


FIGURE 5: Structure modeling the status of a user fronts an elementary action

All the user's actions (elementary or composed) are sorted according to typical compositions: sequential, parallel, alternative, choice, iterative or of-closure. We present below the principle of each of these compositions. We give, after, the overall structure of the human-computer interaction model. For better understanding and clarity, we represent, in our figures, the elementary structure as a block (a gray colored rectangle).

Sequential Composition

The composition of «n» sequential actions reflects the sequence of their execution. The sequential composition of n actions is ensured by combining the place "*end"* modeling the action i, with the place "*begin"* of the action i +1 (Figure 6).

As an example of a sequential composition, we can imagine a situation where the task analysis revealed that the user must perform three actions one after the other. Such an interaction will be modeled by a sequential composition of three elementary structures (modeling three elementary actions).

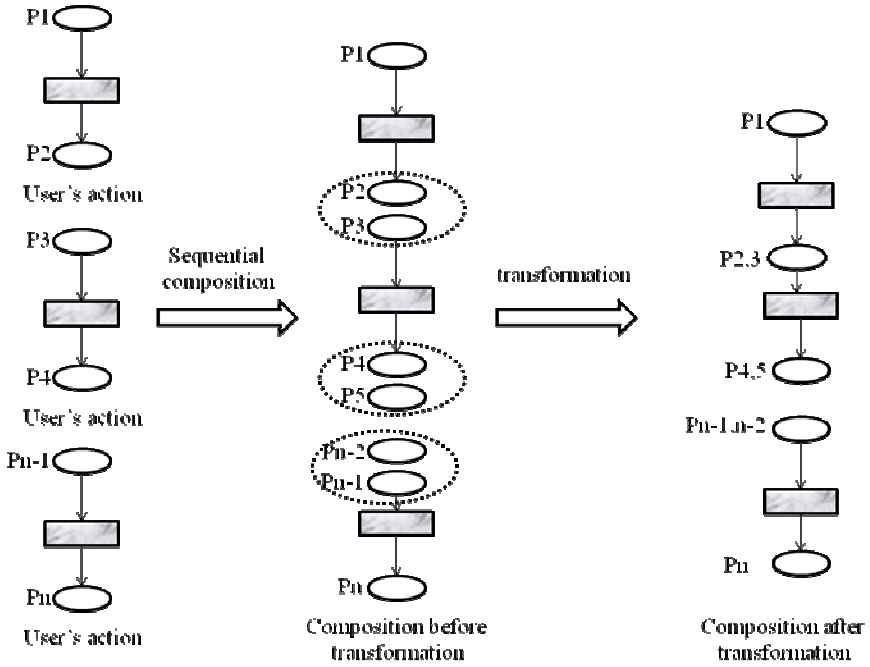


FIGURE 6: Sequential composition

Parallel Composition

The parallel composition expresses the possibility of simultaneous execution. The parallelism is ensured thanks to an input synchronization place. This place activates at the same time all the places of initialization of the parallel actions to be executed.

Note that the effective parallelism can only be done if the actions to be executed do not use the same resources. Otherwise, a partial or complete sequencing would be necessary.

The parallel composition of n networks relatives to n actions involves two compositional structures PAR1 and PAR2.

- The structure PAR1 models a starting synchronization of n networks (Figure 7);
- The structure PAR2 models an arrival synchronization of the n networks (Figure 8).

Obviously, the number of places Pn, in both structures PAR1 and PAR2, must be equal to the number of parallel actions Ai. Thus, to ensure the parallel composition of actions, it is necessary to synchronize the places of entry and those of exit of those actions (Figure 9).

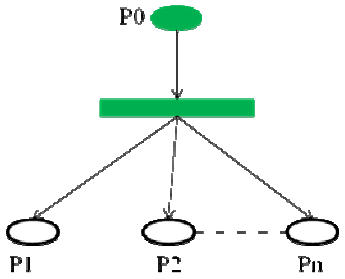


FIGURE 7: PAR1 structure

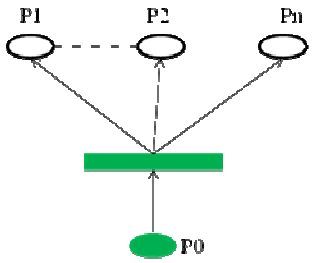


FIGURE 8: PAR2 structure

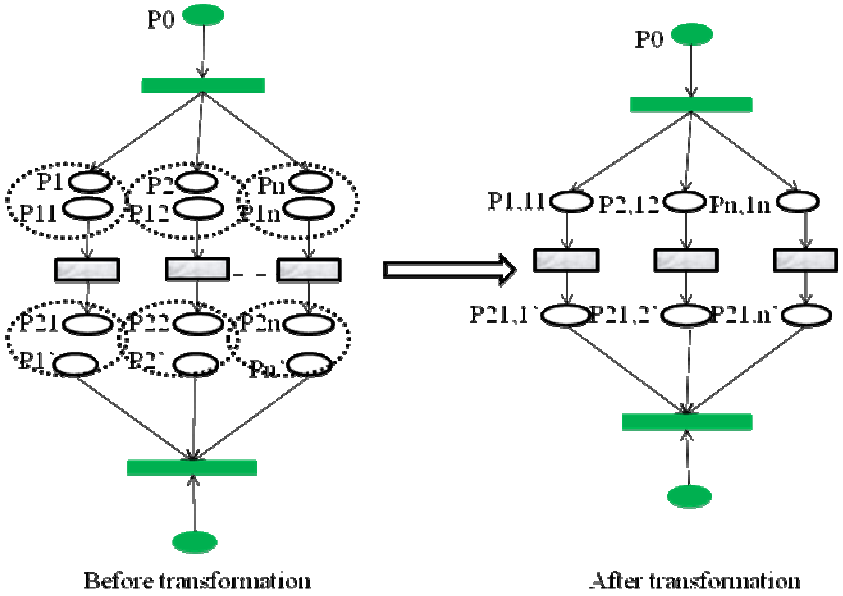


FIGURE 9: Parallel composition

We can consider, for example, a situation where the user has to run two parallel actions. The modeling of this interaction could be achieved by the composition of two parallel branches relative to the actions of launching the user actions.

Alternative Composition

The alternative composition of n actions reflected a performance always exclusive of these actions. To avoid an actual conflict, conditions are associated with transitions to unambiguously determine which action should be executed.

The alternative composition of n networks is realized by composing them sequentially with an ALT structure and merging all the *end* places of these networks. ALT structure allows the validation of a single condition at a time (Figure 10).

ALT structure comprises a set of transitions equal to the number of networks to be composed alternately. These transitions are from the same input place P0. They allow, through the conditions associated with them, without ambiguity to initialize a single PN from the n modeled, which guarantees the absence of actual conflict (Figure 11).

The conditions Ci, i varying from 1 to n, depend on the current status internal to the system.

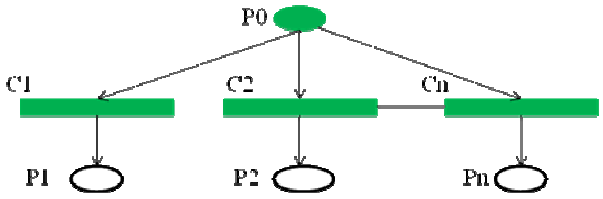


FIGURE 10: ALT structure

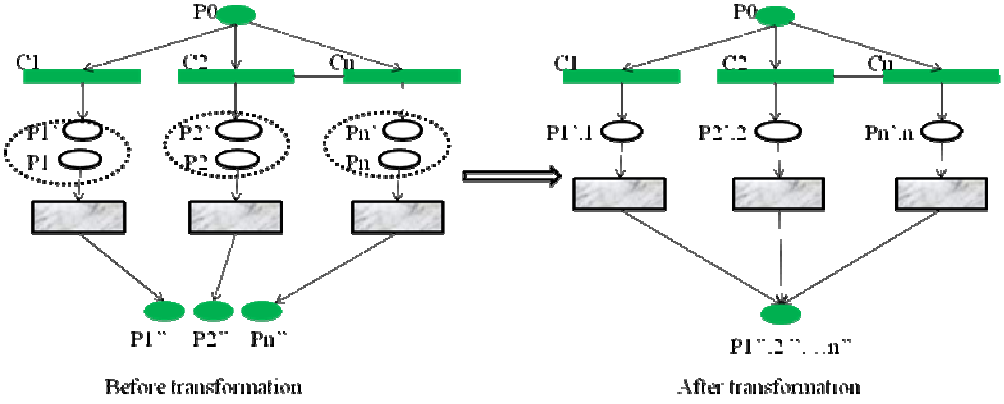


FIGURE 11: Alternative composition

As a simple illustration of this aspect, we can consider the case of a context where analysis has revealed two situations that can never occur simultaneously. For each of these situations, a sequence of user actions is required. Modeling of such interaction is through a composition of two alternative nets each modeling the actions related to a given situation. The associated conditions C1 and C2 relative to the ALT structure of composition will respectively express the occurrence of the two possible situations.

User Choice Composition

There are two kinds of user choice compositions: the exclusive choice and the inclusive choice.

Exclusive choice composition: For the exclusive choice composition, she reminds the alternative composition, with the difference that the decision of the action to execute is not determined by the internal current state of the system but by the user's choice: The user operates through the execution of an elementary action to decide which among the n actions is to be executed. The composition of n Petri nets for an exclusive choice by the user (Figure 13) may be made by composing sequentially an elementary structure of the user's choice decision (Figure 12) with the structure obtained by alternative composition of the n Petri nets.

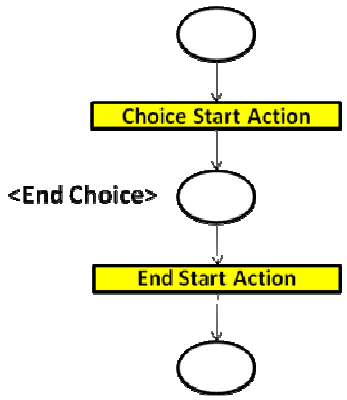


FIGURE 12: Elementary choice of user decision choice

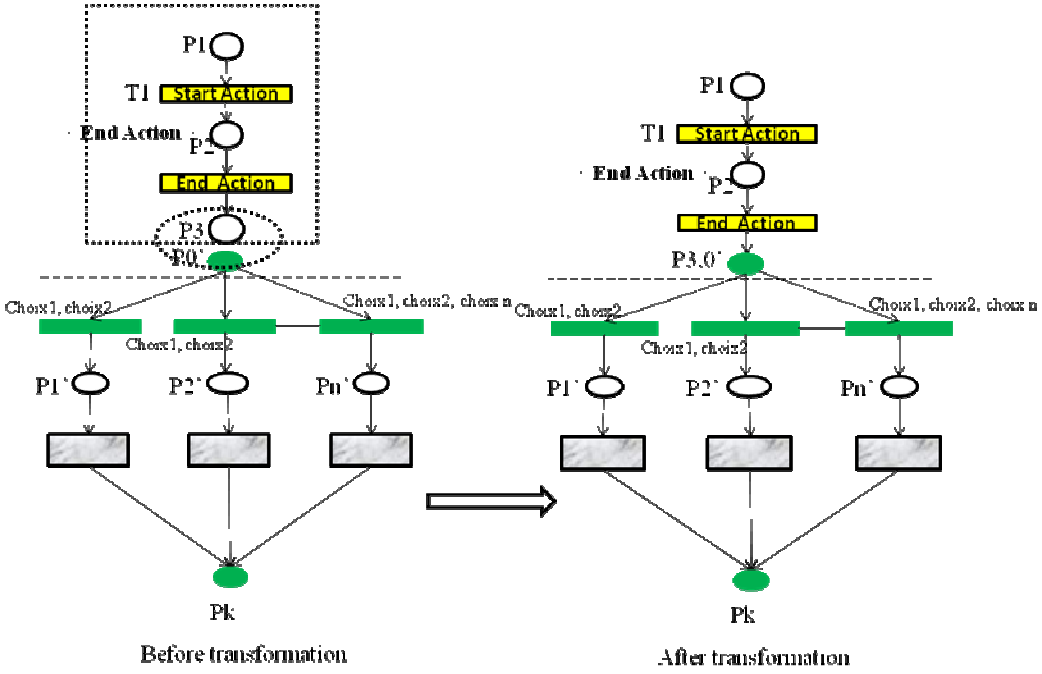


FIGURE 13: Composition of exclusive user choice

As simple case of illustration, we can imagine a situation where the user is forced to choose between only two possible actions: press a button 1 or press a button 2.
Inclusive choice composition: unlike the exclusive choice composition for which the user decides to choose a single action to be executed among n, in the inclusive choice composition, the user can decide to choose a subset of actions to execute among the n (0 or n actions). So, for an inclusive choice, the user intervenes through the execution of an elementary action to decide which actions among the n actions are to be executed, the others will be "short-circuited".

Iterative Composition
The iterative composition of n actions expresses their successive execution with possible iteration. The iteration is subjected to a condition "Iteration" calculated during the sequential execution of the n actions.
The iterative composition of a Petri net is realized by including the Petri net in a structure I of iteration (figure 15). The structure of iteration contains two transitions Ti and Tni. These transitions arise from the same place of entrance. They allow, thanks to the conditions which are

associated with them, to be passed through without ambiguity what guarantees the absence of effective conflict (figure 14).

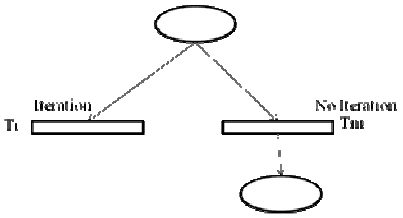


FIGURE 14: Petri I

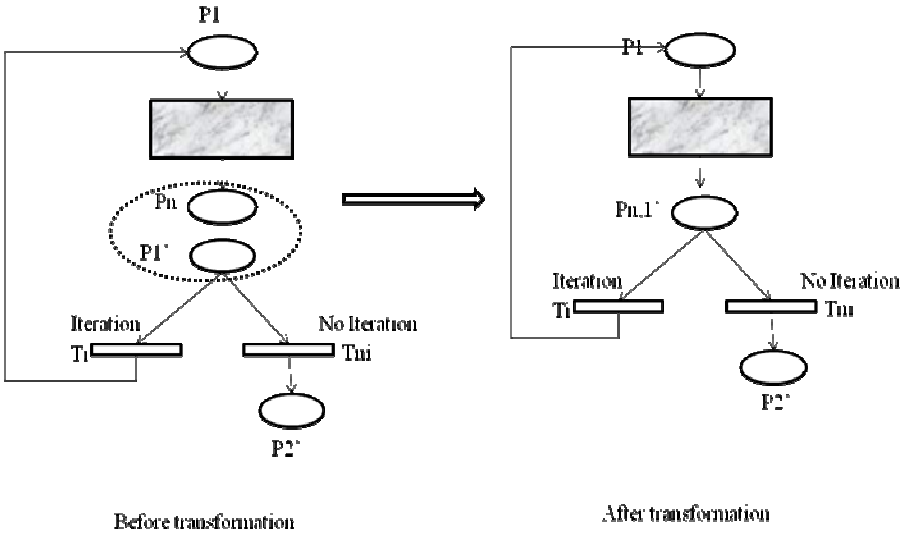


FIGURE 15: Iterative composition

As simple illustration of this aspect, we can imagine a transitional situation of the system which requires the intervention of the user to act on a data by modifying its value gradually by small units. This task will be modeled by an iterative composition of an elementary action relative to the addition of the units every time to the value of the data in entrance.

Closure Composition

The closure composition of a PN translates the looping of this PN. The closure composition of a PN is achieved by including the network in a structure F of closure (figures 16 and 17).

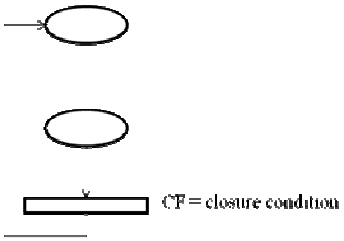


FIGURE 16: Closure structure

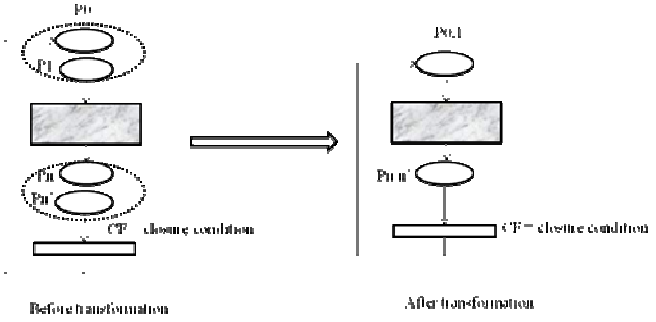


FIGURE 17: Closure composition

The closure composition will be used to build the global model of the human-computer interaction and to express the loop of interaction.

Principle of the Construction of the HCI Global Model

The construction of the global model of a user task is done using the basic structures modeling the different elementary actions of the user. It is based on the application of various operations of composition explained above and its principle lies in the fact that the construction of the model uses only the structures and the composition rules defined. This is important to ensure in advance good properties to the global model obtained.

Note here that normally, any task must be "reset" in order to be rexeceuted again. Building the model of the task must be completed by a closure composition.

Moreover, the enforcement of the rules of composition may lead to Petri nets in which appear obvious simplifications that can be made to reduce model size while preserving its good properties. These simplifications concern, in fact, unstable states. This is the case of (1) the sequential composition of basic tasks not subject to explicit launching conditions or (2) the last stage of the inclusive user choice composition.

Illustration

In Figure 18 we illustrate the process of HCI modeling through the IPN and based on this principle of composition.

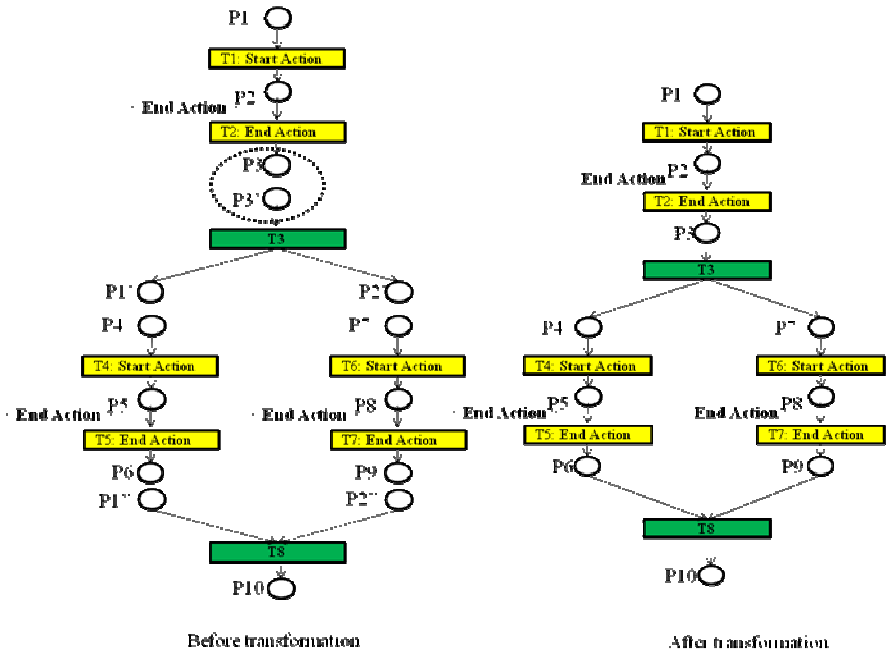


FIGURE 18: Example of a model of Human-Computer interaction

We suppose, in this example that the task of the user is to perform a first action A1 sequentially with two other parallel actions A2 and A3. Each of these actions is modeled by the basic structure presented above, and the global model is build by a composition of three basic structures. The place P2, P5 and P8 models the states of the waiting of the execution of the user actions, A1, A2 and A3.

3. MODELING BY COMPOSITION BASED ON PNML

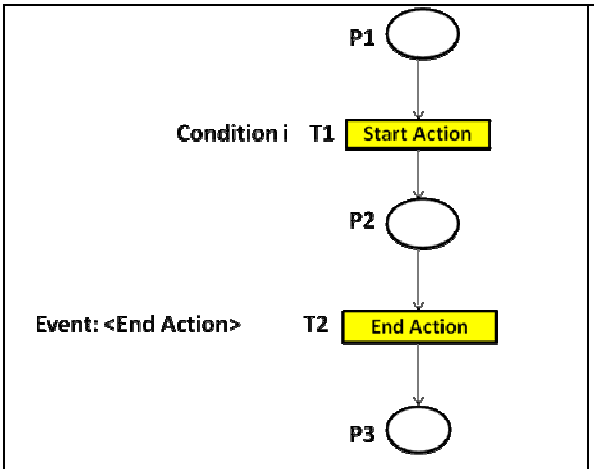
The objective of this modeling by composition is to divide a complex system into a set of sub-systems or modules, simpler, easy to express by verifiable elementary Petri nets. Afterward, we proceed by a composition of these elementary Petri nets, in a recursive way, following the rules of composition indicated above, guaranteeing the maintaining of the verified properties.

The PNML definition of the compositions of elementary Petri nets is made by modeling, at first, the place, the transition and the arc with the syntax PNML (figure 19).

<place id="p1"> <name> <value>p1</value> </name> <initialMarking> <value>1</value> </initialMarking> </place>	<transition id="t1"> <name> <value>t1</value> </name> </transition>	<arc id="a1" source="p1" target="t1"> <inscription> <value>x</value> </inscription> </arc>

FIGURE 19: Definition of a place, a transition and an arc with PNML

Every place possesses two labels to know the tag < name >, to express the name of the place and the tag < initialMarking > to express the initial marking of the place. The representation of a transition is similar to the definition of a place. For the PNML representation of arcs, the source and the target of the node (place or transition) are given as being attributes of the XML element <arc>. PNML requires that every arc possesses a unique identifier.
Using the concept of PNML Modular, each elementary Petri net becomes a module. These basic modules used in their turn, recursively, to compose more complex structures (parallel, alternative, etc.). These complex modules can be used by instantiation in different contexts of design. This leads eventually to obtain a library of modules (simples and complex), which can be used later by the designer to model a complex human-computer system.
For the example of the figure 18, the Petri net is obtained by a parallel composition of two elementary structures, followed by a sequential composition of an elementary structure with the parallel module.
In PNML, a module is defines by the element PNML <module>. This tag contains at once the interface of the module as well as its internal implementation. The interface of the module is labeled by the tag <interface> and contains all the objects of imports and exports of the module.
A PNML module for the structure of Petri Nets of an elementary action of the user will be expressed as shown in the figure 20.

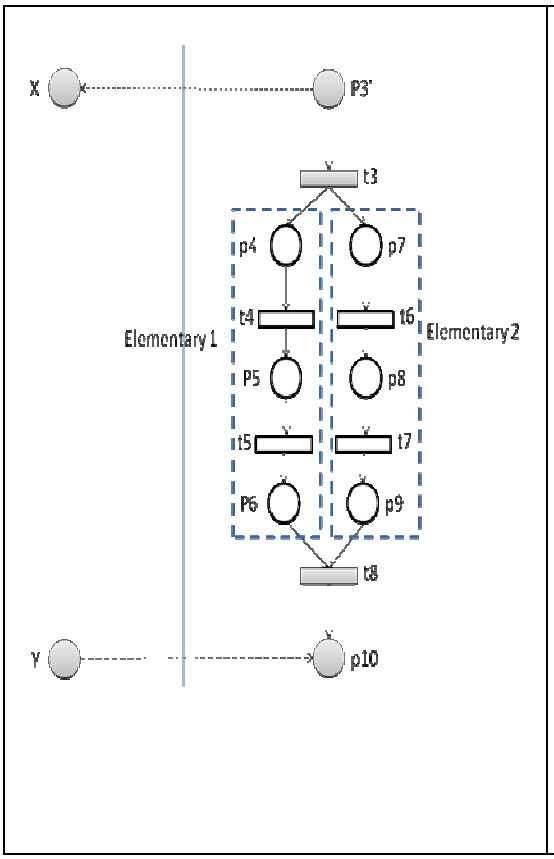



```
<module name="elementary">
    <interface>
        <importPlace id="X"/>
        <exportPlace id="Y"
ref="p3"/>
    </interface>
    <referencePlace id="X"
ref="p1"/>
    <transition id="t1"/>
    <transition id="t2"/>
    <place id="p1"/>
    <place id="p2"/>
    <place id="p3"/>
    <arc source="p1"
target="t1"/>
    <arc source="t1"
target="p2"/>
    <arc source="p2"
target="t2"/>
    <arc source=" t2" target="
p3"/>
</module>
```

FIGURE 20: PNML module for an elementary action

To illustrate our comments, we take back here the example of the figure 18. We propose, below, the modeling PNML of this PN. This composition is made in two steps:

- A first step of making a composition of two parallel elementary actions defined by the PN [p4, t4, P5, T5, p6] and the PN [p7, t6, P8, T7, p9] to get the PN [P'3 , t3, p4, t4, P5, T5, p6, p7, t6, P8, T7, p9, t8, p10]. This PN is defined by a module PNML as shown in Figure 21;
- The second stage which consists in making a sequential composition of two PN: a PN of an elementary action [p1, t1, P2, t2, p3] in sequence with the PN resulting from the first composition, as shows in Figure 22.




```
<module name="parallel">
    <interface>
        <importPlace id="X"/>
        <exportPlace id="Y" ref="p10"/>
    </interface>
    <referencePlace id="x" ref="p3' "/>
    <transition id="t3"/>
    <transition id="t8"/>
    <place id="p3' "/>
    <place id="p10"/>
    <instance id="elementary1"
ref=URI#Elementary>
        <importPlace parameter="p4" ref="p1'
"/>
        <importPlace parameter="p6" ref="p1''
"/>
    </instance>
    <instance id="elementary2"
ref=URI#elementary>
        <importPlace parameter="p7" ref="p2'
"/>
        <importPlace parameter="p9" ref="p2''
"/>
    </instance>
    <arc source="p3' " target="t3"/>
    <arc source="t3" target="p4"/>
    <arc source="t3" target="p7"/>
    <arc source=" p6" target=" t8"/>
    <arc source=" p9" target=" t8"/>
    <arc source="t8" target="p10"/>
</module>
```

FIGURE 21: PNML module for the parallel composition

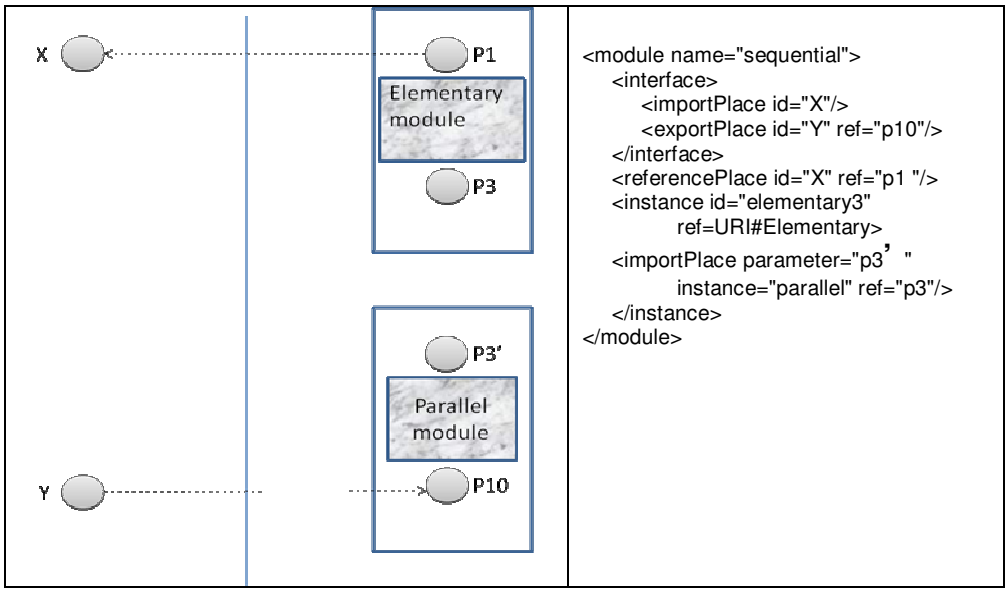


FIGURE 22: PNML module for the sequential composition

The Petri Net modeling the human-computer interaction is defined by this last composition. The following PNML code illustrates this PN (Figure 23).

```
<net id="n1">
    <instance id="global" ref=URI#sequential>
    </instance>
</net>
```

FIGURE 23: PNML module for the overall Petri net

The principle of construction of the PN modeling the human-computer interaction, by composition of the elementary structures guarantees the verification in priori of some properties of the future interface. These properties as well as the process of verification are presented in the following.

4. VERIFICATION OF THE PROPERTIES OF THE MODEL BUILT BY COMPOSITION BASED ON PNML

As the construction of the global model was based on a technique of fusion of the places of the elementary structures, the composition will therefore retain the properties verified on the basic structures. In other words, we can say that, by composing validated elementary structures, the overall model itself is validated. The elementary structure on which the human-computer model interaction is based checks for a number of 'good' properties such as bounding, non-blocking, vivacity and persistence. These properties are therefore guaranteed, by the construction process followed, for the global model of human-computer interaction. This verification of structural properties of Petri nets allows a de facto guarantee of some good properties of the specified interface.

We first present, below, what are these good properties to be checked for the proper functioning of a given interface. We explain, later, the process followed for verifying these properties on the model recommended for the human-computer interaction.

4.1 Properties to Verify

The smooth running of a given human-computer interface can be mainly ensured by the following three properties:

- The absence of blocking: the proposed interface must never jam as a result of any action of the user;
- The availability of services: at some point, the interface must guarantee the availability of certain services relative to the current situation;
- The absence of conflict and the stability: the interface must never pose a situation of effective conflict and it must ensure a certain stability of the various graphic views that it presents. It would thus be necessary to ensure the absence of any state of unpredictable evolution of the interface and there should be no conflict of views in a given running situation.

These good running properties are guaranteed by some structural properties of the Petri nets. Indeed:

- The property of boundedness guarantees a finite number of possible states of the system, which results in stable graphical views at the interface for these different states;
- The property of vivacity reflects the system's potential to reach all the possible modelled services and ensures the absence of any partial or complete blockage, which results in the absence of blocking of the implemented interface and accessibility to the various services, that it offers;
- The property of persistence guarantees the absence of conflict in the network and subsequently the stability of the interface and the absence of confusion or ambiguity with the various states of the system.

We propose below the principle of verification of these structural properties on the human-computer interaction model which is adopted.

4.2 Principle of Verification

The process of construction by composition, studied in this approach, aims at establishing the rules which allow us to systematize this construction but especially to guarantee, in advance, the verification of the characteristic properties of bounding, vivacity and persistence. We show below how the verification of the said properties is effectively guaranteed by this process of construction.

The property of persistence is verified because the elementary structures and the compositions used for the construction of the user task model admit no structural conflict which can lead to an effective conflict.

Let us remember that the process of construction by elementary structures modeling the users actions (ESUA) and built, by successive compositions of Petri nets that we are deliberately going to call «correctly built nets (CBN) ". The last composition of closure transforms the last CBN which is obtained into a user task description model.

A CBN can be defined as follows:

CBN = ESUA \ CBN obtained by applying a unique composition to a set of CBN.

A CBN always has a single place of entrance start and a single place of exit end. It has the following properties:

Pr1: Any registered trademark in the start place of the CBN eventually moves, sooner or later, towards the place of exit at the end.

Pr2: In its progress from the start place the end place, and for any chosen transition in the CBN, we can always find a configuration of values of the variables (internal and\or user) so much so that the progress of the mark leads to passing through this transition.

Pr3: Once the end place reached, all the places of the CBN starting with the start place apart from the end place will have an empty marking.

Consequently, if every CBN satisfies these three properties, the final closure composition of any CBN leads to a model of task which is initialized and which has the property of bounding and of vivacity.

The verification of these three properties is shown below for the elementary structure ESUA and for a sequential composition of two or several elementary structures.

Verification of Properties for an Elementary Structure

Let's remind that an elementary structure ESUA consists of a start place P1, of an action place P2 and of an end place P3. The start of the transition T1 is limited by the condition (condition i).

The passing through transition T2 will take place systematically on occurrence of the event of end of execution of the action associated with place P2. Any mark placed in P1 will move sooner or later towards P2 by the verification of the condition (condition i) and will move on to place P3 starting from the end of this action.

Proposition 1: if any Petri net R satisfies the properties Pr1, Pr2 and Pr3 then the reduction of R by removing an uninitialized ESUA (Figure. 24) preserves reciprocity for the verification of properties Pr1, Pr2 and Pr3.

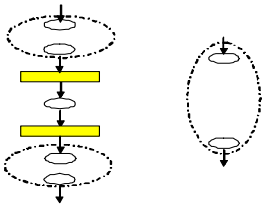


FIGURE 24: Deleting of an ESUA

Evidence: seeing the structure of an ESUA, because it satisfies the properties Pr1, Pr2 and Pr3 and taking into account the precautions of interpretation, the proposal is justified directly by the traditional approaches of analysis by reduction.

Verification of the Properties for a Sequential Composition
According to proposition 1, the verification of properties Pr1, Pr2 and Pr3 for a sequential composition of two or more ESUA is trivial. Indeed, such a composition is achieved by merging the start place of one ESUA and the end place of another ESUA to be composed sequentially. By carrying out successive reductions of ESUA, we end up with one single ESUA which verifies, hypothetically, the three properties.
For the other compositions, the verification of the properties is available in [14].

5. DISCUSSION

As shown in table 1, the PNML extension, proposed here for the HCI modeling, guarantee the exactness of model construction. It facilitates data exchange, allows properties formal verification and assures a simple manner to represent a complex system and exploit it easily.
The models such as ICO, TOOD and XDM assure the exactness of the model construction. But, they haven't studied the problem of data exchange. Also, the models proposed are difficult to exploit and to verify for complex systems.
XML nets and PNML Modular are the source of our idea. But we reproach their manner to construct the PN which allow easily the building of wrong models.

	ICO	TOOD	XDM	XML Nets	PNML Modular	Our proposition (PNML Extension)
Guarantee the exactness of model construction	Yes	Yes	Yes	No	No	yes
Facilitate data exchange	No	No	No	Yes	Yes	Yes
Allow properties verification	It's difficult	It's difficult	No	No	No	yes
Representation of a complex system	Quite Complex and difficult to exploit	Complex and difficult to exploit	Complex and difficult to exploit	Complex and difficult to exploit	Simple and easy to exploit	Simple and easy to exploit
Application domain	HCI modeling	HCI modeling	HCI modeling	Web services composition		HCI modeling

TABLE 1: comparative table of different proposed models based on PN

6. CONCLUSION

We think that for the HCI design, it would be necessary to advance more the assets of approaches of the type MBD. Particularly, those based on formal methods of specification. Indeed, this category of approaches offers a battery of means of verification of HCI which distinguishes it. In our work, we use the formal language of specification of Petri nets for the HCI modeling. For the context-sensitive HCI, we experiment the contributions of OWL. The point shared by these tools is the XML language which unites the OWL and PNML languages

This article concerned a suggestion of extension of the PNML standard. As we can notice, the methodological contribution at the level of the construction of a Petri net model of the HMS, benefiting from advantages of the PNML standard, opens a very promising road towards the formal modeling of the human-computer interaction applied, among others, to the field of mobility.

Our research works continue in this topic and numerous perspectives are under study. We enhance the articulation between the various methods proposed in this approach to end up with a totally integrated approach. We work, also, on the integration of the conditions and the events at the level of the definition of a transition in the PNML description to ensure communication between the model of the interaction and the events of the ubiquitous environment under study. This goes within the framework of the automatic identification of the needs of the users depending on the evolution of the context.

It would also be interesting to implement the approach on complex applications in order to demonstrate the methodological contributions mentioned in this work.

7. REFERENCES

[1] Weiser. M. 1993. "Some computer science issues in ubiquitous computing", *Communications of the ACM* (1993), pp 74-84.

[2] Dey. A.K., and Gregory. D. A. 2000. "Towards a Better Understanding of Context and Context-Awareness". *In Proceedings of CHIA'00 workshop on Context-Awareness, 2000.*

[3] Thevenin. D., et Coutaz. J. 1999. « Plasticity of User Interfaces : Framework and Research Agenda", *International Conference on Human-Computer Interaction; INTERACT'99*, Edinburg, Scotland, 1999.

[4] Thevenin. D. 2001. « L'adaptation en Interaction Homme-Machine : le cas de la plasticité ». *Thèse de doctorat Informatique* préparée au Laboratoire de Communication Langagière et Interaction Personne-Système ({IMAG}), Université Joseph Fourier 238 pages, (2001).

[5] Calvary. G., Demeure. A., Coutaz. J., Daassi. O. 2004. « Adaptation des interfaces homme machine à leur contexte d'usage Plasticité des IHM, La présentation d'information sur mesure »*, Numero Special de RIA; Paris, C. et Colineau, N. (editeurs invites*). Vol 18 (4) 2004. Date de parution: Septembre 2004.

[6] Sottet. J-S., Calvary. G., Favre. J-M., Coutaz. J., Demeure. A., Balme. L. 2006. "Towards Model Driven Engineering of Plastic User Interfaces"*. Satellite Proc. of the ACM/IEEE 8th International Conf. In Models Driven Engineering Languages and Systems*, MoDELS/UML 2006.

[7] Ghiani. G., Paternò. F. 2010. "Supporting Mobile Users in Selecting Target Devices". *Journal of Universal Computer Science*, vol. 16, no. 15, pp. 2019-2037, 2010.

[8] Balme. L., Coutaz. J. 2009. « Ethylene: composants dynamiques pour la mise en oeuvre d□IHM plastiques en informatique ambiante », *IHM 2009,* p 75- 84, Grenoble, France.

[9] Szekely. P. 1996. "Retrospective and Challenges for Model-Based Interface Development",*Bodart, F., Vanderdonckt, J. (eds.). In Proceedings of the Eurographics Workshop, Design, Specification and Verification of Interactive Systems '96,* pp. 1–27, Springer.

[10] Favre. J.M., Estublier. J., Blay-Fornarino. M. « L'ingénierie dirigée par les modèles, au-delà du MDA ». *Hermes, Paris.*

[11] Sottet. J-S., Calvary. G., Favre. J-M. 2005. « Ingénierie de l'interaction homme-machine dirigée par les modèles »*, IDM'05 Premières Journées sur l'Ingénierie Dirigée par les Modèles*,Paris 30 juin, 1 juillet 2005.

[12] Hachani. S., Dupuy-Chessa. S., Front. A. 2009. « Une approche générique pour l'adaptation dynamique des IHM au contexte ». *IHM 2009,* p 89-96, Grenoble, France.

[13] Moussa. F. 2005. « Vers une méthodologie globale de conception et de génération semi-automatique des IHM pour les systèmes industriels »*, Habilitation Universitaire en Informatique,* Faculté des Sciences de Tunis, 2005.

[14] Riahi. M.2004. « Contribution à l'élaboration d'une méthodologie de spécification, de vérification et de génération semi-automatique d'interfaces homme-machine : Application à l'outil Ergo-Conceptor *+ ». Thèse de doctorat,* Université de Valenciennes et du Hainaut-Cambrésis, 2004.

[15] Riahi .I, Riahi .M, Moussa .F. 2011. "XML in formal specification, verification and generation of mobile HCI", *HCII 2011,* 9-14 Juillet 2011, Orlando, Florida, USA.

[16] Shanon. B. 1990. "What is Context?", *Journal for the Theory of Social Behavior,* 1990, Vol.20, pp. 157–166.

[17] Abowd. G., Dey. D., Brown. A. K., Davies. P. J., Smith. N., and Steggles. P. 1999. "Towards a Better Understanding of Context and Context-Awareness": *Proceedings of the 1st international Symposium on Handheld and Ubiquitous Computing*. September 27 - 29, 1999 Karlsruhe, Germany.

[18] Calvary. G., et Coutaz. J. 2002. « Plasticité des interfaces : une nécessité ! », *information interaction intelligence*, Actes des deuxièmes Assises nationales du GDR I3, Nancy, décembre. Toulouse : Cépaduès Editions, pp 247-261.

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

[20] Palanque. P. 1997. « Spécifications formelles et systèmes interactifs : vers des systèmes fiables et utilisables ». *Habilitation à diriger des recherches*, Université de Toulouse I, 1997.

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

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

[23] ABED. M., Ezzedine. H. 1998. « Vers une démarche intégrée de conception-évaluation des systèmes Homme-Machine ». *Journal of Decision Systems*. Vol. 7, pp. 147-175.

[24] De Rosis. P. 1998. "Formal Description and Evaluation of User Adapted Interfaces"*. Int. Journal of Human-Computer Studie*. Vol. 49, 1998, pp. 95-120.

[25] Jambon. F., Brun. Ph., Aït-Ameur. Y. 2001. « Spécification des systèmes interactifs », *In Kolski C, (Ed.), Analyse et Conception de l'IHM, Interaction Homme Machine pour les SI*. Volume 1, pp. 175-206. Paris, Éditions Hermes, 2001.

[26] Palanque. P., Bastide. R. 1995. « Spécifications formelles pour l'ingénierie des interfaces homme-machine ». *Technique et Science Informatique,* vol. 14, n° 4, éditions Hermès, p. 473-500, 1995.

[27] Brun. P. 1998. » XTL : une logique temporelle pour la spécification formelle des systèmes interactifs ». *Thèse en informatique*, Université Paris XI – Orsay, septembre.

[28] Bolognesi. T., et Brinksma. E. 1989. "The formal description technique LOTOS, Introduction to the ISO specification language LOTOS", *Elsevier Science Publishers*.

[29] Hix. D., Hartson. H. R. 1993. "Developing user interface : Ensuring usability through Product process"*, John Wiley Sons, New York, 1993.*

[30] Paterno. F., and Faconti. G. 1992. "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.

[31] D'Ausbourg. B., Durrieu. G., and Rocher. P. 1996. "Deriving a formal model of interactive system from its UIL description in order to verify and to test its behavior". *In Proceedings of DSV-IS'96*. Springer verlag, pp. 104-122.

[32] Paterno. F., Mancini. C.1999. "Designing usuable hypermedia, empirical software engineering", 4(1), pp. 11-42, 1999.

[33] Balbo. S. 1994. « Un pas vers l'évaluation automatique des interfaces homme-machine ». In *Thèse en informatique. Université Joseph Fourier (Grenoble 1), septembre 1994.*

[34] Riahi. M., Moussa. F., Kolski. C. and Moalla. R. 2000. "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 2000, Monastir, Tunisia.

[35] Gruber. T. R. 1993. "Formal ontology in conceptual analysis and knowledge representation", *Chapter: Towards principles for the design of ontologies used for knowledge sharing, Kluwer Academic Publishers, 1993.*

[36] Uschold. M., et Gruninger. M. 1996. "Ontologies: Principles, Methods and Applications"*, Knowledge Engineering Review,* vol.11, n°2, 1996, p. 93-136.

[37] Guarino N. 1997. "Understanding, building and using ontologies". *Int J Human Computer Studies*, vol. 46, 1997.

[38] Lenz. K., and Oberweis. A. 2003. "Inter-Organizational Business Process Management with XML Nets". H. Ehrig, W. Reisig, G. Rozenberg, H. Weber (Eds.): *Petri Net Technology for Communication Based Systems*, LNCS 2472, pp. 243-263, Springer-Verlag, 2003.

[39] Jungel. M., Kindler. E., Weber. M. 2000. "The Petri Net Markup Language", *Proc. 7. Workshop AWPN,* Universitat Koblenz-Landau (2000) 47–52.

[40] Che. H., Li. Y., Oberweis. A., and Stucky. W. 2009. "Web Service Composition Based on XML Nets", *Proceedings of the 42nd Hawaii International Conference on System Sciences.*

[41] Che. H., Stucky. W., and Ju. Y. 2008. "Using XML Nets and Grid Services to support SCOR", *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics,* Kunming, 12-15 July 2008.

[42] Che. H., Mevius. M., Ju. Y., Stucky. W., and Trunko. R. 2007. **"**A Method for Inter-organizational Business Process Management"*, Proceedings of the IEEE International Conference on Automation and Logistics*, August 18 - 21, 2007, Jinan, China.

[43] ISO/IEC, Software and Systems Engineering – High-level Petri Nets, Part 2: Concepts, Definitions and Graphical Notation, *International Standard ISO/IEC 15909* (2007).

[44] Billington. J., Christensen. S., van Hee. K., Kindler. E., Kummer. O., Petrucci. L., Post. R., Stehno. C., and Weber. M. 2003. "The Petri Net Markup Language: Concepts, Technology, and Tools", *24th International Conference on Application and Theory of Petri Nets.* LNCS volume 2679, pages 483-505.

[45] Weber. M., and Kindler. E. 2003. "The Petri Net Markup Language", *Petri Net Technology for Communication-Based Systems-Advances in Petri Nets*, 2003 LNCS volume 2472, pages 124-144.

[46] Hillah. L.M., Kindler. E., Kordon. F., Petrucci. L., Trèves. N. 2009. « A primer on the Petri Net Markup Language and ISO/IEC 15909-2"*. Petri Net Newsletter,* 2009.

[47] Stehno. C. 2002. "Petri Net Markup Language: Implementation and Application"*, PromiseTechnology*.

[48] Barros. J.P., and Gomes. L. 2004. "Operational PNML: Towards a PNML Support for Model Construction and Modification". *Workshop on the definition, implementation and application of a standard interchange format for Petri Nets*, Bologna, Italy, 21-25 June 2004.

[49] Vidal. JC., Lama. M., and Bugarin. A. 2006. "A High-level Petri Net Ontology Compatible with PNML"*, Petri Net Newsletter*, 2006.

[50] Moussa. F., Riahi. M., Kolski. C., Moalla. M. 2002. "Interpreted Petri Nets used for Human-Machine Dialogue Specification". *International journal: Integrated Computer-Aided Engineering (ICAE*), Volume 9, N° 1, 2002, (pp. 87-98). Edition Iopress. ISSN: 1069- 2509.

[51] Moalla. M. 1985. « Réseaux de Petri interprétés et Grafcet ». *TSI de l'AFCET*. Vol. 4 (1), 1985.

[52] Rasmussen. J. 1986. "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).

[53] Norman. D. A., Draper. S. (Eds.), "User Centered System Design: New Perspectives on Human-Computer Interaction". *Hillsdale, NJ: Lawrence Erlbaum Associates (Pointer to Catalog entry in Amazon.com).*