Assessment of Object Use for Task Modeling

Sybille Caffiau[1,2], Patrick Girard[1], Dominique L. Scapin[2],
Laurent Guittet[1], and Loe Sanou[1]

[1] Laboratoire d'Informatique Scientifique et Industrielle, Téléport 2-1 avenue Clément Ader, 86961 Futuroscope Cedex, France
{sybille.caffiau,girard,guittet,sanou}@ensma.fr
2 Institut National de Recherche en Informatique et en Automatique, Domaine de Voluceau - Rocquencourt- B.P. 105
78153, Le Chesnay, France
{Dominique.Scapin}@inria.fr

Abstract. Past research in task modeling suggests the need to introduce objects when using task models for the design of interactive applications. Objects are however rarely included in the task model notations and formalisms. Furthermore, when part of the formalism, their definition is usually informal; and the supporting tool does not generally take them into account for simulation. K-MADe is the first tool that fully uses objects for condition evaluations during task model simulation. This paper presents an evaluation investigating the usage of formal objects with K-MADe. The results show that whilst object concepts seem to be essential in the task model process, their usage and manipulation is not easy.

Keywords: evaluation, task models, objects, K-MADe.

1 Introduction

Designing interactive applications requires a good knowledge of what the users need. One method to gather users requirements is to build task models [1, 2]. Every task model formalism contains different elements to express the user activity such as task categories, scheduling operators and elementary attributes [3]. Many research on task model formalisms pointed out object definition as part of the essential elements in task modeling [4, 5], especially when task models are used to produce interfaces. This work concerns the analysis of the situation [6], the design of interfaces adapted to the context of use [7], or the generation of interfaces from task models [8]. Nonetheless, very few models actually include objects in their formalisms.

One interest of using a task model editor is the ability it offers to validate task scheduling along with the user. In order to facilitate this validation, task model editors contain simulation tools. To reach this aim, objects must be dealt with.

This paper presents an evaluation of the definition and use of task model objects. In order to perform this study, we used K-MADe as a tool support (the corresponding tool of the K-MAD formalism [9]). This tool has been chosen for two main reasons: first the object definition is formal (important aspect to validate task models) and

P. Forbrig and F. Paternò (Eds.): HCSE/TAMODIA 2008, LNCS 5247, pp. 14–28, 2008.

second, it is currently under development thus, the results of this evaluation will be used to improve the tool usability and usage. The first two parts of this paper present the objects used in task models, particularly in the K-MAD formalism and in its associating tool K-MADe. The following parts describe the various steps of the evaluation; going through the goal, participants, procedure, equipment, collected data and, at last, a critical analysis.

2 Objects in Task Models

Task analysis is essential to design interactive applications [10]. In order to facilitate the task analysis process, task models were developed. Due to the wide diversity of task model formalisms and notations, a comparison of the different systems [2] and their components was conducted [1]. This second comparative study highlights the presence of objects in the majority of formalisms to introduce domain models using references, or to embody them in task models. In this paper, we focus our study on task model formalisms that embodied objects.

As stated by Limbourg [1]: "A tool clearly facilitates the task modeling activity, hiding the model notation from the analyst and helping him or her capture it." p137. Moreover our evaluation necessitates the use of a tool. Thus, we looked at the use of objects in task model tools. Five task model formalisms (and their associated tool) correspond to these two criteria: CTT [11] (CTTE), Diane+ [12] (TAMOT), GTA [4] (EUTERPE), MAD* [13, 14] (IMAD) and K-MAD [15] (K-MADe). We will briefly present the use of objects in the tools before comparing them. Then, we describe in further details the use of objects, before comparing the tool we chose for our study, K-MADe.

2.1 Formalismes Using Objects

CTT (CTTE). CTTE objects [16] are a task property. They are characterized by: a **name** (string); a "**class**" among *string, numeric, object, description* or *position*; a **type** among *perceivable* (object presenting any information or allowing action of user) and *application* (intern in the system); an **access mode** (*only reading* or *modification*); a **cardinality** among *low*, *median* and *high*; and the **platforms** where the object is represented. To our knowledge, no documentation describes in details the concepts of class and cardinality. According to our use of the tool CTTE, we associate the need of the cardinality characteristic with the generation of interfaces [8] based on CTT diagrams. However, the CTTE simulation tool does not take into account objects and we only found some documentation relating to the use of objects for interface generation.

Diane+ (TAMOT[1]). The task model Diane+ integrates objects, named **data**, and uses them to define **conditions**. However, in the associated tool: TAMOT, the only editable condition is the pre-condition, expressed in the form of a string.

GTA (EUTERPE). In EUTERPE, objects are first class components. They are characterized by a **name** (string); a **list of attributes** (each attribute is composed of a

[1] http://www.ict.csiro.au/staff/Cecile.Paris/IIT-Track-Record-Past-Projects/Projects/Isolde/Tamot/Index.htm

name (string) and a **value** (string)); and a **list of users** (the users who can manipulate the object). The users are defined as **labeled agent**. Relations can be defined between agents and objects (*owner, create, destroy, use/inspect, change*). An **agent** is defined by a **name** (string); a **type** (*individual, organization* or *computer system*); and a **role** (a *set of tasks* performed by an agent). Moreover, EUTERPE allows the definition of **events**. These are composed of a **name** (string) and the **set of tasks** they are linked with (task set).

MAD* (IMAD). Two types of object are present in MAD*. They correspond to the object-oriented notions of class (**abstract objects**) and instance (**concrete objects**). Each **object** is composed of a **name** (string), a **number** (the *ergonomic number* corresponding to its place in the task tree (integer)), a **list of attributes**. The abstract object attributes are characterized by a **name** (string) and a **type** (*string, boolean, integer*) and concrete object attributes by a **name** (string) and a **value**. Some characteristics are addressed to abstract objects as a meta-class (generalizing link); a sub-class (specializing link); a condition of instance numbering (restriction to one instance). However, in IMAD the two types of objects are not differentiated. The user cannot give a value to IMAD object attributes.

K-MAD (K-MADe). K-MAD allows the definition of **entities** that characterize the environment of the user. These entities either represent what s/he handles or what influences the course of his/her activity. The various types of these entities are: **users** (set of users implicated in the activity); **events** (set of events that can be triggered or caused by the activity); **objects** (set of concepts handled by the user).

As for MAD*, objects can be abstract or concrete. Whilst abstract objects are composed of the characteristics of the objects that are manipulated by users in real world, concrete objects are instances of abstract objects. Each object possesses attributes: abstract attributes (belonging to abstract objects) are their characteristics. Concrete attributes, belonging to concrete objects, aim at associating a value to each characteristic defined by an abstract attribute. These objects are used to define pre-conditions, post-conditions and iteration conditions. K-MAD includes groups of concrete objects, in addition to the definition of the users and the involved events.

In this paper, we present an evaluation of the use of all these elements referred by the term “entities”.

2.2 Comparison of Objects in Tools

Table 1 synthesizes the different paradigms used in the five task model tools. Only two of them define the notions of *events* and *users*; EUTERPE and K-MADe. In these two tools, they are associated with the tasks.

With the exception of TAMOT (which does not contain *data* of the model Diane+), all the tools contain the concept of objects and they can be split in two categories. First, the tools considering objects as task attributes (as CTTE), and then the tools considering objects as first class component of the formalism (as EUTERPE, IMAD and K-MADe).

Moreover, in CTTE, a particular object attributes is its cardinality. It is used to help designer define the interactive element presenting this object. In addition, perceivable objects may be a table or a window… thus this tool associates interactive objects to

Table 1. Comparison of concepts in task model tools

Tool	Events	Users	Objects	Conditions
CTTE			*Object*	pre-condition (String)
TAMOT				pre-condition (String)
EUTERPE	Event	Agent	Object	pre-condition (String)
				post-condition(String)
IMAD			Class	pre-condition (String)
				post-condition(String)
K-MADe	Event	Users	Abstract Object	pre-condition (Formal Expression)
			Concrete Object	post-condition (Formal Expression)

tasks. The introduction of these elements (cardinalities and perceivable objects) in task model formalism illustrates the link between objects and interface presentation.

This definition is close to a system point of view, whereas in all others tools, object concepts aim to be closer with to ergonomic point of view. Whilst objects defined in CTTE are concrete (a value is associated to the object since its definition), IMAD does not allow giving a value to object attributes, staying in an abstract level of definition. This level of definition freezes the manipulation of task model objects.

All the formalisms include pre-conditions associated to the tasks. Their validations are mandatory for the execution of the tasks. Then, to allow the validation of task models by the user and thus, the verification the task scheduling (using simulation), these conditions need to be computed. In order to compute them, definitions of objects and conditions have to be formal. Among all the tools, whilst K-MADe allows these formal definitions, all others define conditions using non-computing string.

Due to this possibility of computation of expressions (for instance during the simulation of task models) using these objects, the degree of K-MADe object definition is limited (i.e. object is composed of predefined types) while objects in EUTERPE may be composed of other objects.

After observing two types of object definition in task model formalisms, we can define three groups of tools according to the type of object definitions. Firstly, a group of tools with a low level of formal definition (i.e. containing definitions allowing no verification (as IMAD, TAMOT and CTTE)). Secondly, the medium group, containing EUTERPE, that does not contain formal objects but defines formal relationships between them and tasks. Last, the more formal tool, K-MADe allows formal definition of objects and conditions, which allows using objects during simulation. As we stated in the introduction, this possibility seems essential for our purpose.

3 Presentation of the Tool K-MADe

K-MADe (K-MAD environment) [9, 15] has been developed to model, manipulate, and evaluate the K-MAD formalism. It implements the different characteristics of the K-MAD model. We used it to perform our evaluation of the usage of objects in task models. In section 3.1, we give a general presentation of K-MADe. Section 3.2 outlines the specificities to use objects in the tool.

3.1 General Presentation

The K-MADe tool is targeted towards people wishing to describe, analyze, and formalize activities of human operators or users. It allows the creation of task models concerning non-computerized or computerized experiments, real-world or simulated situation, on the field or in laboratory. Whilst all kinds of profiles are possible, this environment is particularly intended for ergonomists and HCI specialists. Due to the wide range of user's background and skills, the tool allows different levels of description, from simple graphics to detailed mathematical expressions using the following available tools:

- A graphic editor of the K-MAD task model. It uses direct manipulation techniques to build, handle and cancel tasks (label 1 in Figure 1).
- Editors of task characteristics (see the list above). Label 2 in Figure 1 indicates one of the three representations it provides.
- An editor of abstract objects, users, events and concrete objects. Objects can be added, modified and removed. The editing and removal of objects implies the modification of all associated objects. *Sheets* (label 3 in the Figure 1) allow to access these object definition editors.
- An editor of expressions for pre-conditions, post-conditions and iterations. The tool is able to check the grammar of expressions, and to evaluate them.
- A simulator that allows animating task models.
- Tools for analysis of task models (statistical, coherence, queries…).
- A tool for printing task trees and task characteristics.

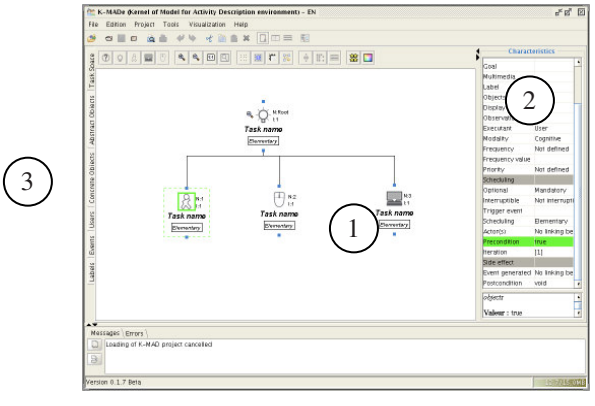

Fig. 1. The main window of K-MADe tool

3.2 Objects in K-MADe

Several K-MADe components are dedicated to the different entities we previously mentioned. We classify them into two groups; components for editing and components for usage.

Entity edition. Each K-MAD entity is defined using different windows. Editing events and users is equivalent to textually label them and eventually to add a description to them. Contrary to these basic and informal definitions, editing objects is more detailed. Two different windows allow the definition of the two object types; one for abstract objects and one for the concrete ones. The Figure 2 presents the window for editing abstract objects. Types of object attributes (label 1 in Figure 2) are defined among usual programming types (boolean, string, integer). Moreover, concrete objects are accessible only through groups of abstract objects, thus in the abstract object editor, groups are editable (label 2 in Figure 2).

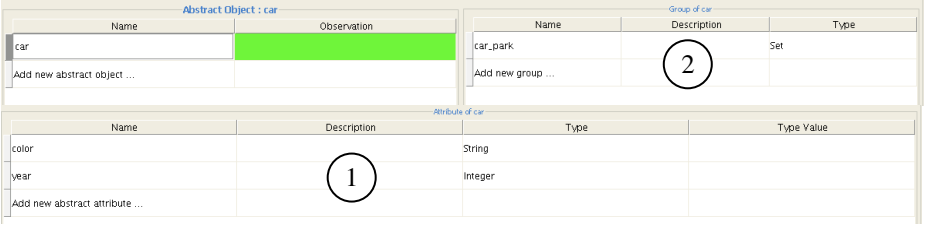

Fig. 2. The K-MADe abstract object editor

The Figure 3 from [15] shows relationships between abstract objects, concrete objects and groups.

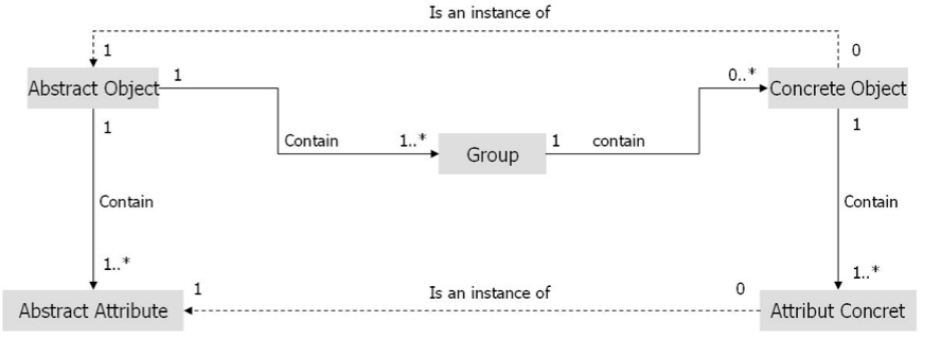

Fig. 3. Relationships between abstract objects, concrete objects and groups

Object usage. K-MADe entities are used to characterize task events (triggered and generated), conditions (pre, post and iteration) and authorized users. The designer chooses from the set of defined ones to associate users and events with task. Conditions (using objects) are edited using the calculator based on B semantics [17]. A special kind of calculator is dedicated to each condition type. Figure 4 shows the calculator for the pre-condition edition. Once edited the conditions can be used to help simulate and consequently validate task models.

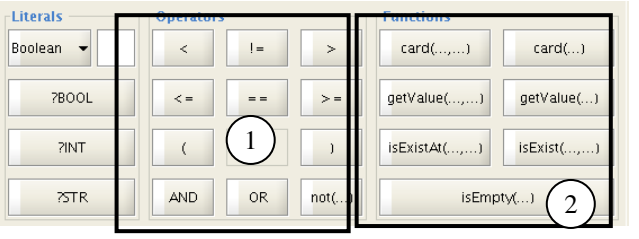

Fig. 4. Pre condition editor calculator

4 Goal of the Study

The study presented in this paper aims to evaluate the use of entities (objects, events and users) in modeling the users' activities. Whilst K-MADe is addressed to users with different skills (computer scientists, ergonomics…), the participants to our study are students in HCI. The target of our evaluation is using task models for application design. K-MAD task model formalism was developed to present the different steps of the task analysis, staying in the task analyst point of view. Thus, the entities are not described using developer vocabulary. For example, the term "class" is not used to identify the abstract object concept. In order to better understand why entities are/can be used, we focus on two aspects: the role they play for users and what consequences their use have on task models. Following these aspects, we separated our study into two evaluations.

The first evaluation aimed at defining schemas of modeling processes focusing on the processes where entities are edited and used. In the second evaluation, we investigated the difficulties related to these concepts: the understanding of the tool, and the definition of models using K-MADe.

5 Participants

All participants were students in their fourth year of French university. There were split into two groups, each participant performing only one of the two evaluations. The first group was composed of 48 bio-informatics students, and the second one of 20 computer science students (studying computer science since their first year of university). Only one computer science student (participant in the second evaluation session) is not French speaker. However, they all attended the same HCI course. This course focuses on user-centred design and where task modeling is presented. The K-MAD formalism was explained in details in the lecture (approximately 4 hours) and students practiced task modeling using K-MADe before the evaluation (approximately 6 hours, performing some task models checked by tutors). Then, even if they are not modeling experts they were more trained to use a task model notation than ergonomic task model experts [18].

The second part of this course was focused on evaluation (basic concepts of evaluation and main methods used in evaluation [19]). As students play the role of evaluators, the protocol applied in this survey is used as an example in order to

facilitate their future evaluation workload. However, as this study was their first practical evaluation, their participation was limited to the observation and its annotation. Moreover, their notes were completed with other data.

6 Evaluation Method

In order to perform this evaluation, we used a widely used evaluation technique [19]: real-time expert observation of subjects using the tool. In this part, we will present the experimental procedure, the directives that were given to the participants, and finally, the method used to complete the expert's evaluation.

6.1 General Organization

The two evaluations were performed with a gap of one month between the first and last evaluation. Each evaluation followed the same process. All students were paired. During the first session, one student acted as task model designer (using K-MADe, labeled *user*), while the second acted as the expert (named *observer*). They reversed roles during the second session. Each session lasted one hour and a half with a fifteen minutes break between sessions. The activity to model was the same for all students and it was introduced in French at the beginning of the sessions.

6.2 User Work

The user had to model the activity of completing a volley-ball game marking sheet. Instructions for this activity were given at the beginning of sessions. They were composed of the official instructions of the French Federation of Volley-Ball (FFVB) and two examples of marking sheets (completed and non-completed ones). K-MADe was used to model the tasks to perform.

6.3 Observer Work

During modeling, observers insured that their user verbally described their modeling process, and annotated what they observed concerning the use of the tool by the user (hesitations, exploration in several parts of the software without actions and so on.). In order to help observers in their evaluation, we gave them observation sheets (illustrated in Table 2). These sheets were mainly composed of a three columns table corresponding to the three types of information recorded for each observation:

- The type of the observation among a set of defined categories (user goal (G), tool functionalities (F), functionality utilization (FU) and information (I)).
- The observation in textual form.
- The time of observation.

6.4 User-Logs and Questionnaires

In order to complete the observers' notes and the task model performed, we used two others types of data.

Table 2. Observation sheet example

Type	Observation	Temps
FU	The principal window is not accessible ("simulation" is noted on but the simulation window is not accessible too). => launch again K-MADe	14h32
G	looking for the object definition	14h34 14h37
F	user does not understand the signification of the button with shell-hole	14h40

User-Logs. To complete the observations realized during the evaluation, the users in the first evaluation used a version of K-MADe with sneaks. These ones allowed keeping track of user's actions using timestamps and produced a text-file (the user-log). Particularly, this log indicates when the user enters and exits each K-MADe tool (task space, abstract objects, condition editions (pre, post and iteration)…). Figure 5 shows an example of information in this file.

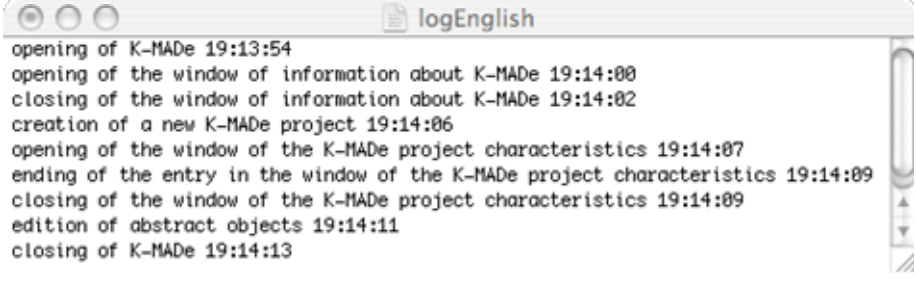

Fig. 5. Illustration of data recorded in the user-log

Questionnaires. Participants were asked to complete a questionnaire in French about the use of objects in the second session of the evaluation. This questionnaire was composed of five questions on definitions, object deletions and conditions.

7 Data

As each of the two evaluations tried to reach different goals, we did not collect the same data for both experiments. In this section, we present the results of each study. Table 3 resumes the data gathered according to the session and the evaluation goal.

7.1 Collected Data

First evaluation. The first evaluation session aimed at analyzing the task modeling process, particularly when the K-MAD entities are defined and manipulated. In order to obtain this information, we used user-logs and notes from the observers. These two

types of information allow the collection of two complementary data. While the observer is focused on the user usage, what her/his goals are and how s/he conceptually model, the user-logs give information on how the K-MADe components are used. Using timestamps on both data, we can determine how users use K-MADe tool components.

Moreover, we requested of each student to exploit their notes and the user-log to write an evaluation report. It includes the modeling process of the observed people, his/her use and usage of the tool, and an analysis of the resulting model. Whilst the produced documents were then readable and quite organized, models, observer notes and user-logs were also collected for analysis.

This first evaluation session helped to determine when K-MADe entities were used in the general modeling process. However we did not collect any precise information about their usage, the second session aims to answer this question.

Second evaluation. As for the first evaluation, the user's behavior was reported in the observer notes and a document was written to report clearly their observations. However, user-logs do not gather information about entity usage thus we did not use them for this session. In order to analyze K-MADe entity usage, we considered two types of data: the models and the questionnaires. Verification of entities in the resulting models indicates the degree of understanding of the object concept. Questionnaire analysis (associated with the student report analysis) aims to inform us on the difficulties and the needs of using objects.

Table 3. Data gathered and its goal

Session	Data	Goal
1	- user-log	user activity when modeling
	- observer-student notes	student-user comportment
	- student exploitation document	writing report of the notes
	- model	verification of student analysis
2	- observer-student notes	student-user comportment
	- student exploitation document	redaction of their notes
	- model	validation of object definition and usages
	- questionnaire	object concepts

7.2 Selection of Data

During the first evaluation we collected one folder per user. It included the observer notes, the user-log, the observer exploitation document and the task model. This study aimed at gaining some understanding on the modeling process. The data used to deduce users modeling process was mainly taken from user-logs. This file was automatically generated without any technical problem. However, we did not want to use these user-logs without taking into account the context (reproduced in the observer-student notes and exploitation document). Two of the folders were not complete and therefore were not included into the analysis. We ended up considering 46 out of the 48 folders in our analysis.

The data used for the second evaluation included all the information in the folders, we could therefore only consider the fully-completed folders. However whilst for the

first analysis the observer notes, the student exploitation document and the task models were only used to help us give a context to the user-log data, for the second one they were essential. During this evaluation process, we observed that the only non-French speaker student could not understand all the directives (this observation was confirmed when he ought to complete the questionnaire). He was therefore not considered in the analysis. The second part of our evaluation is based on 19 complete folders.

8 Object Definition and Use in Task Modeling Process

Prior to identify the intervention of objects in the task modeling process, we observe that some students did not define objects. Indeed, 26% of users (12/46) of the first session evaluation did not try to define (or use) any K-MAD object. However, we cannot precisely identify why. Two reasons may explain the absence of these elements in task model process: the limited duration of the experiment, or the non-assimilation of object concepts. Student notes and reports did not allow us to identify the main reason. Six participants indicated that the sessions were not long enough but others (6/12) did not give any relevant information on the subject.

From the 34 remaining folders, we identify three main schemas followed by user to perform task models. The most used ones (43,75%) are divided into two steps. Firstly, the user composes the task tree (decomposes tasks). Secondly, s/he iteratively edits entities and associates them with tasks. Steps of the second most used schema (28%) are sequential. The user performs the task decomposition prior to define all entities, and then associate them with the tasks (using conditions). Moreover, an incomplete process (followed by 22% of user-students) is composed by the first two steps of the second schema. The last schema is the iteration of the second one. The Figure 6 resumes the fist and the second schemas.

These observations give us some understanding on the place objects have in the task modeling process. As an example, the concurrent definition of objects and task tree composition indicates that the user associates objects and tasks. On the contrary, when the definition and the use of objects are separated with the task tree composition, we can deduce that the user defines objects only to use them on conditions. Therefore, objects bear the role of associating properties and tasks for some users.

```
task tree composition;
While (time is not finished)
{
   edit objects;
   define properties using
them;
}
```

```
task tree composition;
edit objects;
define properties using
them;
```

Fig. 6. Schemas of student task modeling process

From the data gathered in both evaluation sessions, we want to identify what are the elements used during the task modeling process. Table 4 shows how many

students define each task model component. According to these results, few users integrated the concepts of *events* (20% in the first evaluation and 26% in the second) and *users* (20% and 10,5%) in the task modeling process. On the contrary, the major part of the first evaluation users and all user-students of the second defined the objects (abstract and concrete objects) to model the activity.

Table 4. Users defining each of K-MADe elements

	Event	User	Abstract Object	Concrete Object	Group	Pre	Post	Iteration
1	20%	20%	74%	60%	74%	40%	40%	20%
2	26%	10,5%	100%	100%	100%	84%	89,5%	84%

The definition of event and user objects is textual. Associating them with tasks is easy using selection among the defined elements. On the contrary, the abstract and concrete objects are composed of several concepts. Thus, the difference of the use between these two types of concepts (composed and non-composed ones) cannot be explained by the level of difficulties of the definition.

However, Table 4 indicates also the proportion of users using pre, post and iteration conditions. In the first evaluation session, these three types of object manipulation were widely used in the task modeling process (84% defined at least one pre-condition, 89,5% defined at least one post-condition and 84% defined at least one iteration condition). Prior to edit these conditions, the user needs to define the objects (abstract objects, concrete objects and groups). Then, the objects may be defined only to allow the definition of conditions. Therefore, users did not conceived objects as a part of tasks but as a way to define conditions.

From these numbers, we observe that for the majority of students, it is natural to define objects in order to complete the semantics of scheduling operators. As an example, a volley-ball game ends when one team wins at least 3 sets, no matter of the score of the other team. This condition for the end of a game cannot be expressed using only scheduling operators. All students naturally defined it using objects and conditions.

The study of the proportion of definition of K-MADe concepts of the two sessions shows a difference between the two groups. In the first evaluation, 26% of users do not use any entity. All participants followed the same lecture then, the explication of this gap cannot be found in this teaching. However, these groups do not have the same background. Numbers shown in Table 4, clearly indicate that formal object definitions are easier to be used for computer scientist students.

9 Object Usages

Students use K-MADe in order to define and use entities in task modeling process. Our evaluation aims to understand conceptual and procedural usages of objects. All analysis presented in this part are based on the data gathered in the second evaluation.

9.1 Conceptual Usages

Even if second evaluation students integrated entities in task modeling and used them (in pre, post and iteration conditions), 58% of them noticed that some of these concepts were not understood. These difficulties did not affect the concepts of events and users, used less (26% use events and 10,5% use users) but that were easy to use.

On the contrary, concepts of abstract objects, concrete objects and groups represented respectively 72%, 64% and 27% of difficulties of understanding. Likewise, the use of conditions was not understood by 10,5% of users. 87,5% of users that did not understand the concept of abstract objects (resp. concrete objects), did not also understand the concept of concrete objects either (resp. abstract objects).

The definitions of these two types of objects are very closed, thus the difficulty of using objects seems to be the link between them. This analysis is supported by another observation. Whilst all students indicate that they wanted to edit and use objects, one of them could not describe his definition of objects (abstract, concrete objects and groups).

As we said before, in K-MADe, the manipulation of the concrete objects necessitates the use of groups (label 2 in Figure 2). The role played by the group concept represents a difficulty of conceptual understanding for 67% of users of the second evaluation (we did not collect the point of view of the participants of the first evaluation). Firstly, they indicate that they do not understand why the groups are required for the definition of concrete objects showing the non-understanding of the relationship between concrete objects and groups (shown in Figure 3). Secondly, K-MAD does not allow the definition of the number of elements in groups. Then, the users regret the need to define a group to use only one concrete object.

9.2 K-MADe Object Usages

Our evaluation highlights difficulties from the use of K-MADe to manipulate concepts. Some of users need to edit the K-MAD concepts in several steps. Therefore, 42% of users indicated the need to edit again at least one abstract object and 37% at least one concrete object. Concerning the definition of groups, the proportion is more important because nearly one student out of two (47%) did not define every group at the first attempt.

We did not gather any information about the modifications done during the edition of objects. However, 42% of users indicated that they wanted to delete at least one abstract object, 63% at least one concrete object, and 47% at least one group. These needs seem to indicate that modifications are important on objects. Moreover, they confirm the conceptual understanding difficulties shown preliminarily.

K-MAD (abstract and object) objects are composed of attributes. K-MADe allows to define them with a name and to associate a type of value (or the value for concrete object). The available types of value (label 1 in Figure 2) respectively are boolean, integer and string. These types are used by 58%, 89,5% and 42% of users. The variety of available types are widely used, as only one 37% of users used a unique attribute type to define objects (string or integer).

As users are computer scientists, they are familiar with attribute types. Therefore, they do not find difficulties in understanding and using type attributes. Due to their

background, they feel necessary to define other types of attributes such as date, hour or defined objects. Moreover, they indicated the non-understanding of the need of groups to define the concrete objects (58%).

Naming attributes, objects and groups requires the respect of a particular syntax without stresses, underscores and spaces. However, the tool does not indicate these syntactic rules and automatically changes spaces in the names (when the users put them). As these modifications are done automatically (without either any intervention of the users nor any indication), the consequent errors during the condition computations were not understood by the users. 30% of the users indicate they showed this error type during the modeling process.

Then, the last observation concerns the manipulation of objects via calculators. To allow the manipulation of objects and the combination of them, there are operators (label 1 in Figure 4) and defined functions (label 2 in Figure 4). The use of this second type of elements induced some problems. In the tool, there are not any explications either about the sense of the functions, nor to precise what are the order and the type of parameters

10 Conclusion and Future Works

In this paper, we presented an evaluation of the use of the object concept using a task model tool; K-MADe. The data gathered during the evaluation supports the theoretical idea that objects are part of task models. In order to formally express conditions (and complete the scheduling of the task decomposition), the introduction of objects in the modeling process appears intuitive for the participants of our evaluation. However whilst the necessity of the definition of these task entities does not cause any conceptual difficulty for participants, using them is more difficult.

Users indicated some lacks in the tool that influence their task model process: for example, the limitation of object attribute types (no date or hour format) or the impossibility to define an object composed of others (as EUTERPE proposes).

In addition, we observe two major difficulties in the usage of task model objects. Firstly, some of them, the events and the users, are not used and then, do not seem to be understood by users. In the lecture these concepts were presented along with the others thus the lack of usage seems due to the concepts themselves, either to their presentation in the tool or to their definition. No evaluation data allows to precise this fact. Secondly, defining formal conditions using objects is not user-intuitive. Reasons of this difficulty may be the use of the calculator (non-intuitive) or the representation of objects (that need to be naturally manipulated). In order to improve this usage, we need to modify the calculator and presentation of objects.

However, the participant's skills in computer science do not allow us to generalize our observations to all users. Moreover, as we shown before, a minor difference of skills considerably modifies the usage of objects (see Table 4). In order to gain a broader point of view, the same type of evaluations with other background participants has to be performed.

Last, this evaluation aimed at understanding the usage of objects on task modeling process. However in K-MADe, objects are taken into account in the simulator tool then, a research plan of experimental studies will be performed investigating the role of objects in the task model validation step.

References

1. Limbourg, Q., Vanderdonckt, J.: Comparing Task Models for User Interface Design. In: Diaper, D. (ed.) The Handbook of Task Analysis for Humain-Computer Interaction, pp. 135–154 (2004)
2. Balbo, S., Ozkan, N., Paris, C.: Choosing the Right Task-modeling Notation: A Taxonomy. In: Diaper, D., Stanton, N.A. (eds.) The Handbook of Task Analysis for Human-Computer Interaction, pp. 445–466 (2004)
3. Lucquiaud, V.: Sémantique et Outil pour la Modélisation des Tâches Utilisateur: N-MDA. Thesis. Poitiers, 285 (2005)
4. Van Der Veer, G.C.: GTA: Groupware Task Analysis - Modeling Complexity. Acta Psychologica, 297–322 (1996)
5. Dittmar, A., Forbrig, P.: The influence of improved task models on dialogues. In: CADUI, pp. 1–14 (2004)
6. Van Der Veer, G.C., Hoeve, M., Lenting, B.F.: Modeling Complex Work Systems - Method meets Reality. In: Green, T.R.G., Canas, J.J., Warren, C.P. (eds.) 8th European Conference on Cognitive Ergonomics (EACE), INRIA, Le Chesnay, pp. 115–120 (1996)
7. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A unifying reference framework for multi-target user interfaces. Interacting With Computers 15/3, 289–308 (2003)
8. Paternò, F., Santoro, C.: One model, many interfaces. In: Kolski, C., Vanderdonckt, J. (eds.) Computer-Aided Design of User Interfaces (CADUI 2002), Valenciennes, Frances, pp. 143–154 (2002)
9. Baron, M., Lucquiaud, V., Autard, D., Scapin, D.: K-MADe: un environnement pour le noyau du modèle de description de l'activité. In: Robert, J.-M., David, B. (eds.) IHM 2006, Montréal, Canada, pp. 287–288 (2006)
10. Hackos, J.T., Redish, J.C.: User and task analyis for interface design. Wiley, New York (1998)
11. Paternò, F.: Model-Based Design and Evaluation of Interactive Applications (2001)
12. Tarby, J.C., Barthet, M.F.: Analyse et modélisation des tâches dans la conception des systèmes d'information: la méthode Diane+. In: HERMES (eds) Analyse et conception de l'IHM, interaction pour les Systèmes d'Information, vol. 1, Paris (2001)
13. Gamboa, R.F., Scapin, D.L.: Editing MAD* task description for specifying user interfaces, at both semantic and presentation levels. In: Harrison, M.D., Torres, J.C. (eds.) Eurographics Workshop on Design, Specification and Verification of Interactive Systems (DSV-IS 1997), Granada, Spain, pp. 193–208 (1997)
14. Scapin, D., Bastien, J.-M.C.: Analyse des tâches et aide ergonomique à la conception: l'approche MAD*. In: Kolski, C. (ed.) Analyse et conception de l'I.H.M. / Interaction Homme-Machine pour les S.I., Paris, France, vol. 1 (2001)
15. K-MADe electronic reference, `http://kmade.sourceforge.net/`
16. Paterno, F.: ConcurTaskTrees: An Engineered Notation for Task Models. In: Diaper, D., Stanton, N.A. (eds.) The Handbook of tTask Analysis for Human-Computer Interaction, pp. 483–501 (2004)
17. Lano, K.: The B Language Method: A guide to practical Formal Development (1996)
18. Couix, S.: Usages et construction des modèles de tâches dans la pratique de l'ergonomie: une étude exploratoire
19. Nielsen: Usability Engineering (1993) ISBN 0-12-518405-0