Proposition d'un noyau et d'une structure pour les modèles de tâches orientés utilisateurs

Vincent Lucquiaud

I.N.R.I.A.
Domaine de Voluceau – Rocquencourt, B.P. 105
78153, Le Chesnay Cedex, France
Vincent.Lucquiaud@INRIA.fr

RESUME
L'analyse de la tâche de l'utilisateur est une étape primordiale du processus d'ingénierie des systèmes interactifs. Les nombreux modèles de tâches actuellement disponibles peuvent laisser supposer que cette étape est correctement supportée. Ces modèles, pour peu qu'ils soient outillés, exposent un large spectre de capacités – évaluations, interrogations, simulation, etc. La tentation de les utiliser conjointement suppose des interopérabilités. Or, même au niveau syntaxique cela demeure impossible. De plus, suite aux divers travaux effectués dessus, des problèmes apparaissent, tant sur la finalité des caractéristiques des tâches que sur leurs documentations ou sur le maintien de leur cohérence. Pour tenter d'y remédier, une approche modulaire est proposée afin de structurer ces modèles et répondre à ces exigences. Enfin, un module central, le noyau, est exposé il contient les données minimums pour décrire l'activité et ce indépendamment de toute idée de spécialisation.

MOTS CLES : Analyse des tâches utilisateur, outil, modèles, noyau.

ABSTRACT
User task analysis is a critical step in the design process of interactive systems. The large set of user task models available today may lead to assume that this step is well supported. These models, which possess sometimes an associated tool, expose a wide spectrum of proficiency such as evaluation, interrogation, or simulation. Temptation to use them jointly presume interworkings. But, even at the syntactic level it remains impossible. Furthermore, following various work, problems appear as well on task's caracteristic, as documentation or coherency. To try to remedy, a modular approach is proposed to structure these models and answer these requirements. Thus, we will expose here our central module, the core, which contains minimal datas to describe the activity and this, independently of any idea of specialization.

CATEGORIES AND SUBJECT DESCRIPTORS: D.2.1 [Software Engineering]: Requirements/Specification – Methodologies.

GENERAL TERMS: Design, Langages

KEYWORDS: User task analysis, tools, models, kernel.

INTRODUCTION

Dans le domaine de la conception des interfaces interactives, les modèles de tâches sont utilisés afin de tenir compte des besoins de l'utilisateur final avant même la conception. Dans cette optique, l'analyste décrit l'activité de l'utilisateur par un arbre de tâches qui décompose les tâches en sous-tâches. Les modèles comme Diane+[13], CTT[10], MAD[11] ou GTA[15], qui ont respectivement pour outil Tamot[14], CTTE[1], IMAD et Euterpe[2], présentent aussi des capacités à travers ces outils : capacité à évaluer, analyser ou simuler.

Chacun de ces outils possède des capacités différentes, ce qui nous incite à vouloir en utiliser plusieurs simultanément afin de profiter d'un maximum de capacités. Cependant, ceci nécessite une interopérabilité entre ces outils. Outre les échanges de données entre les formats des outils, il existe un impératif préalable : les données décrites par l'un doivent être compréhensibles par l'autre.

Un modèle possède un intérêt tout particulier quand il est outillé, cet article se concentre donc sur les outils. Il examine la faisabilité du passage d'outil à outil. A cette fin, la première section est orientée sur le contenu de ces outils. Puis nous expliquons pourquoi et comment nous sommes arrivés à proposer une structure modulaire pour ce type de modèle. Enfin, nous présentons les données contenues dans ce noyau pour la modélisation de la description de l'activité.

POUVOIR D'EXPRESSION DES OUTILS

Le pouvoir d'expression est le reflet des capacités syntaxiques d'un formalisme [5]. Nous passons en revue tous les concepts présents dans les outils à savoir : les tâches, les utilisateurs, les évènements, et les objets de l'utilisateur. Les objets sont rapidement abordés, ils constitueraient à eux seuls un autre article.

La tâche est le point central, par conséquent, les autres composants sont d'une manière ou d'une autre reliés à celle-ci. C'est pourquoi les tâches sont présentées à la fin.

L'utilisateur

Les utilisateurs sont les exécutants des tâches. Euterpe les définit avec deux composants à savoir les rôles et les agents. Un rôle est une collection significative de tâches accomplies par un ou plusieurs agents. Il est inspiré de TKS [6]. Il est défini par un nom, une liste de tâches et d'agents. Un agent quant à lui est une entité active (personnes, groupes ou composants logiciel). Il est caractérisé par un nom et un type qui est un énuméré : {Individuel, Organisation, Ordinateur}, c'est-à-dire que le champ contient une des valeurs de cet énuméré.

Dans CTTE, il y a une description de l'activité (un arbre) par acteur et une description générale. Cet acteur est caractérisé par un nom. Dans IMAD cela n'existe pas, bien que, dans MAD* un utilisateur est défini par son statut au sein de l'entreprise, par la fonction qu'il remplit, par son expérience dans la tâche et par les compétences requises dans celle-ci. Si les deux premières caractéristiques sont bien liées à l'utilisateur, les deux autres sont liées à la tâche. Or un utilisateur n'est pas associé à une seule et unique tâche.

L'objet de l'utilisateur

Tous les outils, sauf Tamot, permettent de décrire plus ou moins bien les objets utilisés par l'utilisateur. Ces objets peuvent avoir une influence sur le déroulement de l'activité. Cette influence se traduit par une pré ou post condition et éventuellement une itération dont l'expression fait intervenir ces objets. Néanmoins, aucun outil ne les décrit de façon suffisamment formelle pour prétendre les utiliser par la suite.

L'évènement

Seul Euterpe en permet la description. Un évènement indique un changement d'état interne (les objets, tâches, agents) ou externe (intempérie, provision électrique). Euterpe ne caractérise l'événement que par un nom, un commentaire et un lien sur une tâche. Ce modèle ne spécifie pas comment l'événement est créé et par qui [15]. De plus, rien n'est dit sur leurs gestions au niveau du déroulement de l'activité.

La tâche

Les attributs de la tâche sont groupés en quatre groupes : le premier a trait aux informations de la tâche, le second aux caractéristiques de la tâche, le troisième à l'ordonnancement local et le dernier à l'ordonnancement global de la tâche.

Informations de la tâche. Cette section regroupe l'ensemble des attributs informationnels de la tâche (tableau 1). Tous les champs sont du texte, il n'y a donc aucune sémantique.

Outils	nom	numéro	commentaire	objectif	média
Euterpe	String		String	String	String
IMAD	String	string	String	String	
CTTE	String		String	String	
Tamot	String		string		

Tableau 1 : bilan sur les informations de la tâche.

Si tous les modèles attribuent un nom et un commentaire pour la tâche, seul IMAD gère automatiquement une numérotation, tandis que Euterpe peut associer un média à la tâche. Exprimés dans le langage de l'utilisateur, ces attributs sont peu propices à servir au niveau de la capacité génératrice (ce qu'on peut faire faire).

Caractéristiques de la tâche. Elles regroupent les propriétés des tâches n'ayant aucune influence dans l'ordonnancement de la tâche (tableau 2). Pour la plupart, leurs syntaxes sont exprimées à l'aide d'énumérés, de chaînes de caractères (string) ou de listes ([] dans le tableau). Cette dernière peut contenir plusieurs valeurs.

Le choix des colonnes est réalisé selon la syntaxe, hormis la fréquence pour IMAD et CTTE qui divergent dans leurs expressions. Il est à noter que seul CTTE peut définir une tâche comme étant abstraite et précise la plate-forme sur laquelle la tâche agit. La « pertinence » dans Euterpe ne permet pas de définir une tâche comme étant importante et facultative. De plus, tout comme IMAD, les auteurs ne précisent pas dans quelles conditions les colonnes « mentale » et « moteur » sont applicables. Une tâche automatique n'est ni mentale, ni moteur. Remarquons aussi que le feedback est vu différemment selon l'outil, pour Tamot c'est un champ alors que pour CTTE c'est un type de tâche. Le transfert de données pour cette catégorie engendre une perte d'informations.

Outils	Colonne 1	Pertinence	Mentale	Moteur	Fréquence	Colonne 6	Plate-forme
Euterpe	***Type*** : {complexe, élémentaire, interaction, ø}	{normale, critique, dangereuse, optionnelle, ø}	{vrai, faux, ø}	{vrai, faux, ø}	String		
IMAD	***Modalité*** :{manuelle, automatique, interactive}	{Importante, relativement importante, secondaire}	{sensori-motrice, cognitive}		{Elevée, moyenne, faible}		
CTTE	***Catégorie*** : {utilisateur, système, interaction, abstraite}				{basse, moyenne, haute}	***Type*** : { comparaison, feedback, …}	[PDA, Bureau, CellPhone, …]
Tamot	***Style*** : {manuelle, automatique, interactive}					***Feedback :*** Alphanumérique	

Tableau 2 : bilan sur les caractéristiques de la tâche. ***En gras*** : le nom de l'attribut dans les outils

Ordonnancement local de la tâche. Ces propriétés ont une influence sur l'ordonnancement de la tâche considérée indépendamment de ses sous-tâches (tableau 3). L'utilisation de ces propriétés implique, à des fins d'évaluations ou de simulation, qu'il y ait une sémantique. Or en général, la valeur des champs est informelle.

Outils	Durée	Nécessité	Interruption	Itérative
Euterpe	String	Ambigu		
IMAD	début, fin, durée	{facultative, obligatoire}	Booléen	
CTTE	min,max, moyenne	Optionnelle	opérateur d'ordonnancement	Booléen
Tamot		Booléen		***Cardinalité*** min, max

Tableau 3 : bilan sur les caractéristiques locales.

Concernant la durée, rappelons qu'une description est globale contrairement au scénario qui repose sur un parcours particulier dans l'arbre. Par conséquent, définir le temps aussi précisément semble plutôt être de l'ordre du scénario et non de la description. Par exemple, donner le début d'une tâche en terme de durée a un sens dans le cadre d'un scénario mais est difficilement compréhensible dans le cas d'une description.

La durée, de même que l'interruption et l'itération, sont définies différemment. Il y a donc incompatibilité. La nécessité dans Euterpe est ambiguë (c.f. pertinence), alors que dans CTTE et Tamot il y a une valeur par défaut, ce qui signifie que la valeur peut ne pas être explicite. Ajoutons que CTTE permet, avec un booléen, d'identifier une tâche comme étant coopérative et que IMAD a un champ priorité.

Un autre aspect est présent dans ces modèles. Il s'agit des conditions d'exécution des tâches en lien avec les objets de l'utilisateur. Le tableau suivant en présente un bilan :

Outils	précondition	postcondition	Etat initial	Etat initial
Euterpe	String	String	String	string
IMAD	string	String	String	string
CTTE	String			
Tamot	String			

Tableau 4 : bilan sur les relations objets et tâches.

Tous les modèles permettent de décrire une précondition, c'est-à-dire une configuration dans laquelle la tâche est autorisée à s'accomplir. Les postconditions sont quant à la configuration des objets une fois la tâche accomplie. Cependant, aucun ne les utilise au moment de la simulation, car ces conditions ne reposent pas sur un langage formel utilisant les objets. Bien que le modèle MAD* présente une « grammaire », rien n'a été trouvé sur son fonctionnement (assignation d'une instance à une classe ? modification des attributs ?). De plus, les instances n'existent pas dans IMAD ce qui empêche toute évaluation.

Concernant les états (initial et final), ceux-ci induisent l'idée qu'il y a une transformation sur les objets. Or il n'en est rien à en juger les définitions s'y rapportant. Notamment, il est mentionné qu'un objet dans l'état final ne doit pas contenir de contraintes, [12] pages 16. Comme le souligne cette même note de recherche « suite aux entretiens, il semblerait que la frontière entre état et condition ne soit pas toujours clairement établie pour l'analyse » page 3. Cette remarque est par ailleurs illustrée par un exemple.

Ordonnancement global de la tâche. Cette partie est dédiée aux opérateurs d'ordonnancement lesquels fixent, selon deux types distincts, la décomposition des tâches : soit entre les sous-tâches (IMAD, Euterpe, Diane+), soit entre les tâches sœurs (CTTE). CTTE utilise les opérateurs LOTOS[4]. La signification n'étant pas la même, le tableau ne peut pas comporter de critères. Le problème est de savoir s'il existe une équivalence entre ces deux types de décompositions sachant que la description résultant de cette transformation doit rester compréhensible.

Outils	Opérateurs d'ordonnancement
Euterpe	Ø
IMAD	{séquentiel, parallèle, simultané, et, ou, alternatif, élémentaire}
CTTE	{pas d'ordre, choix, concurrence, concurrence*, activation, activation*, désactivation, interruption} *avec passage d'information
Tamot	terminal, séquentiel, parallèle, et, ou, XOR

Tableau 5 : bilan sur les opérateurs d'ordonnancements.

Les modèles de tâches présentent une collection d'opérateurs d'ordonnancements dont certains ne se limitent pas à exprimer la nature de cette décomposition entre la tâche mère et ses sous-tâches. Certains expriment le fait qu'il y a des échanges de données. Par ailleurs, cette précision est souvent soit une redite soit un fait qui ne peut être exprimé par le modèle lui-même. En effet, si le modèle est capable de décrire les objets, il doit être possible à l'aide des conditions de déterminer quels échanges il y a, dans le cas contraire -les modèles ne décrivent pas les objets- à quoi sert cette information, comment est-elle par la suite utilisée ? Il en est de même pour la simultanéité qui induit que les sous-tâches sont exécutées par plusieurs utilisateurs. Si les utilisateurs sont décrits, il doit être possible de déterminer cette simultanéité à partir des personnes qui accomplissent les sous-tâches. Si les utilisateurs ne sont pas descriptibles à quoi sert ce type information ?

Il en résulte que l'échange de données, entre les différents modèles de tâches ne suffit pas pour tirer profit des capacités de chacun. En outre un modèle dont le contenu reposerait sur le plus petit dénominateur commun à l'ensemble des données ne permettrait pas de recouvrir complètement un seul des modèles. Ce regroupement, matérialisé par le projet DOLPHIN[9], qui propose de transférer des données entre les différents modèles, pose comme nous l'avons vu des problèmes de complétude, de syntaxe et de sémantique insolubles. En conséquence, nous cherchons un noyau pour un modèle de description orienté utilisateur qui serve de point d'entrée à d'autres

modèles recouvrant une autre partie du cycle de développement, UAN[7] par exemple. Avant toute chose exposons la raison d'être de ce noyau.

STRUCTURE DU NOYAU

En premier lieu, nous exposons les raisons qui nous ont amenées à proposer un noyau pour un modèle de tâches orientées utilisateurs, puis, nous expliquons la manière dont celui-ci a été conçu avant de montrer les avantages liés directement à la structure du noyau.

Constats

Tout d'abord, arrêtons-nous un instant sur les études relatives à ces modèles de tâches (Diane, Euterpe, GTA, MAD), sur la façon dont sont gérées leurs documentations ainsi que sur les stratégies employées pour les actualiser en cas d'évolution. L'usage de ces modèles, dans le cadre de la recherche, est de les examiner suivant un objectif propre aux besoins de l'étude en question. Cette étude peut être la gestion des interruptions, l'évaluation prédictive de la charge de travail, la génération d'une maquette de la future interface, etc.

Bien souvent la réponse à une étude conduit les auteurs non pas à actualiser le modèle utilisé, mais à greffer de nouvelles caractéristiques et règles, et ce, sans en étudier les conséquences sur l'ensemble de celui-ci, ou tout au moins, à distinguer les caractéristiques propres à la description de celles liées à une autre étude. Le passage de MAD à MAD* a créé des ambiguïtés, par exemple, concernant les opérateurs ET, OU et le caractère obligatoire de la tâche (associés ils n'ont plus de signification). La présence d'un attribut n'a de sens que dans certain cas de figures. Par exemple dans IMAD, l'interruption, la condition déclenchante et la priorité n'ont de sens qui si la tâche est pourvue de l'opérateur PAR ou SIM [12].

La réponse à un objectif peut prendre éventuellement une autre voie : par la création d'un autre modèle ou version. Le modèle GTA a, à ce jour, au moins trois outils implémentant trois versions différentes, et qui plus est, non compatibles. Chacun utilise ses propres données selon ses besoins avec au passage des ajouts pour satisfaire les objectifs fixés. La continuité dans l'évolution du modèle n'est pas assurée. C'est en tout cas comme cela que les travaux, à l'aide des publications, nous sont présentés.

Une raison, peut être, à cela est le manque de documentation précise et exhaustive sur les données manipulées, leurs raisons d'être et les règles (sémantique) qui composent le modèle. Par conséquent, un article pris en référence représente trop souvent qu'une sous-partie du modèle. La raison est que l'article aborde plus l'objectif de recherche que le modèle lui-même. Il n'existe ainsi que trop rarement un document de référence pour un modèle donné qui contienne une partie descriptive, une partie sur les règles et une autre sur la sémantique du modèle. Il semble donc difficile de déterminer la grammaire et la sémantique d'un modèle à travers un unique article. Avec deux, il n'est pas rare de trouver des conflits et de définir exactement les caractéristiques du modèle. Ce problème a été rencontré plusieurs fois dans la section précédente.

Une autre explication, concerne le suivi des évolutions des modèles. Un modèle reste théorique avant d'être pratique. L'ambiguïté, difficilement décelable dans un contexte théorique, l'est beaucoup moins quand la finalité de l'outil implique obligatoirement une sémantique formelle. Encore faut-il que le logiciel soit fait dans un but similaire en termes d'exigences formelles en ne se limitant pas à la description. Notons néanmoins que le suivi des évolutions d'un modèle à travers son outil rencontre moins de problème, peut-être est-il plus « naturel » d'assurer une compatibilité descendante, son élaboration étant un processus lui-même itératif.

Pour répondre à ces problèmes dans le suivi des modèles, nous proposons de construire un noyau de description de l'activité afin que celui-ci constitue la référence à un modèle. Autrement dit, il peut constituer un point de départ pour les études suivantes. Ce noyau est constitué d'entités et de champs considérés comme minimum pour une description de l'activité, avec tout de même une orientation interface. Il reste à expliquer la manière dont celui-ci a été conçu. Ceci fait l'objet de la partie suivante.

Construction du noyau

Dans la pratique, ces modèles de tâches ont tous recours à une description de l'activité de l'utilisateur, celle-ci constitue soit la finalité même du modèle, soit un besoin préalable à l'obtention de résultats dérivant de cette dernière. Néanmoins, et ce quelle que soit la nature de ce recours, les concepts, mais aussi les attributs manipulés par l'ensemble de ces modèles, forment pour la majorité d'entre eux des ensembles disjoints (c.f. première section).

Ceci révèle une corrélation entre les données manipulées et les objectifs liés à la description des modèles, mais aussi et surtout, la non-unicité de ce qu'est la spécification d'une activité. Dans ce contexte, comment prétendre à la détermination d'un noyau, étant donné qu'il expose de façon intrinsèque ce qu'est la description de l'activité, alors même que l'étude du pouvoir d'expression des modèles met justement en défaut la définition de cette complétude ?

Dans le premier cas, c'est-à-dire lorsque la finalité du modèle est l'analyse de l'activité, ce recours manie des attributs et des concepts dont la présence est pour le moins subjective étant donné le manque de justification. Il est donc difficile de définir les exigences en matière de complétude dans ce cas. Dans l'autre cas, c'est-à-dire pour les modèles ayant un objectif en lien avec la description de l'activité, et qui représente la grande ten-

dance actuelle grâce à l'apparition d'outils, ils permettent éventuellement l'étude des champs et des concepts selon leurs usages et leurs objectifs. Le lien entre un objectif, les champs et les concepts est trop rarement formulé. Dans le premier cas la définition est empirique dans le second, elle est plus rationnelle, bien qu'elle ne concerne qu'indirectement la description de l'activité.

Le noyau, au sens où nous l'entendons, a pour objectif de contenir les concepts et les caractéristiques de base pour aspirer à décrire une activité aussi générique que possible. Ces caractéristiques doivent être suffisamment générales et pas uniquement tournées vers une spécialisation particulière. Ce noyau a un impératif, être stable et une exigence, être non-ambigu. Pour le construire nous nous inspirons des modèles de tâches orientées utilisateurs et outillés (GTA, MAD, CTT, Diane) en nous appuyant notamment sur leurs éléments communs.

C'est pour ces raisons, notamment, que le noyau ainsi créé ne prétend pas introduire une définition de ce que doit être la complétude en matière de description de l'activité. Néanmoins, il fixe des bases concernant la structure d'un tel modèle. Enfin, il répond en conséquence à la mise en place d'une stratégie pour actualiser et maintenir un noyau stable et non-ambigu. La partie suivante expose la structure du modèle qui découle de ce noyau.

Structure du noyau

Le noyau tente de regrouper les données jugées comme étant essentielles à la description de l'activité et propose un procédé d'actualisation et de vérification pour les études futures. Précisons qu'à ce stade, la structure sous jacente est présentée en dehors de toute idée d'architecture logicielle. Ceci ne constitue pas l'objet de cette partie qui est d'expliquer la structure du point de vue du modèle.

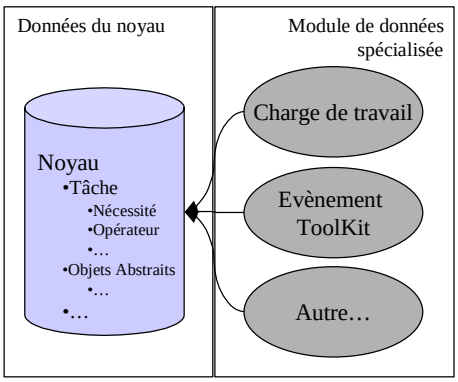

Figure 1: Stratégies d'évolutions du N-MDA.

Nous proposons de ne retenir dans le noyau du modèle que les caractéristiques en lien direct avec la description de l'activité. Les autres caractéristiques ajoutées sont considérées comme étant propres à un autre objectif et feront l'objet d'un module à part qui les contiendra. La figure 1 illustre ce procédé. Ce module sera par exemple dédié à la charge de travail ou à la description des actions sur l'interface. Par ailleurs, si pour remplir un objectif le besoin d'ajouter des caractéristiques, des concepts ou d'étendre le pouvoir d'expression du noyau se fait ressentir, une vérification sur la compatibilité avec ce dernier est plus que souhaitable.

Le principe est de distinguer les attributs ou plus généralement les concepts n'ayant pas de relation directe avec la description de l'activité mais constituant une spécialisation. Le noyau permet de structurer les modèles de tâches en module et de rendre plus lisible leurs spécifications. A charge ensuite pour l'utilisateur d'utiliser les modules suivant ce qu'il souhaite réaliser. Par cette méthode, il est déchargé de certaines caractéristiques qui parfois, de part leur nombre, ne facilitaient pas l'utilisation et la lisibilité du modèle.

NOYAU DU MODELE DE DESCRIPTION DE L'ACTIVITE.

Le ***N***oyau du ***M***odèle de ***D***escription de l'***A***ctivité contient un module, dédié à la description des objets des utilisateurs qui n'est pas présenté ici. Puisque toutes les entités se lient avec les tâches nous présentons d'abord les utilisateurs et les évènements. Dans tous les cas, nous donnons les règles de constructions, de contraintes ou d'existences conditionnelles quand elles existent.

L'utilisateur

Cette section a pour but de recenser les acteurs (personne ou groupe) intervenant dans l'activité. Seules les caractéristiques propres à l'utilisateur sont décrites ici. Celles liées à la tâche, quand celui-ci en est responsable, constituent une relation de lien. L'utilisateur est donc décrit par un nom, un statut (au sein de l'activité) et par la fonction que celui-ci remplit. Ces champs sont tous des alphanumériques, il n'y a donc pas de contrôle possible sur leur contenu. Seules les caractéristiques propres à l'utilisateur sont acceptables à ce niveau.

Concernant la spécification des utilisateurs, elle est régie par une unique **règle de contrainte** : deux utilisateurs ne peuvent avoir le même nom.

L'évènement

La notion d'événement est utilisée car il est bien distinct de l'objet manipulé qui est, lui, contrôlé par l'utilisateur. Un évènement est un aléa qui n'est pas du ressort de l'utilisateur mais qui influence le déroulement de son activité. La notion d'événement permet de faire un pont entre ce qui est décrit dans l'activité et ce qui n'est pas considéré dans la dite activité.

Un évènement est caractérisé par un nom et une source. La source indique si l'origine de l'événement est liée à l'activité de l'utilisateur ou s'il est complètement extérieur. C'est un énuméré {provoquer, incontrôlable}. Par exemple pour une tâche d'impression, l'événement « plus de papier » est « provoqué » à partir de cette tâche contrairement à l'évènement « alarme » qui peut subve-

nir à tout moment (il n'est pas lié à une tâche). Les événements ont aussi une **règle de contrainte** : deux événements ne peuvent avoir le même nom.

Les tâches

Le noyau décrit-il les tâches prescrites ou les tâches réelles ? Encore faudrait-il distinguer en terme de spécification ce que cela induirait. Dans la mesure où les données rentrées par l'analyste ne peuvent pas être contrôlées pour faire cette distinction, cette précision n'a pas de sens.

Cette section présente le pouvoir d'expression de la tâche. Elle est scindée en quatre sous-sections : les informations de la tâche, les caractéristiques de la tâche, l'ordonnancement local et enfin l'ordonnancement global de la tâche.

Informations de la tâche. A la lecture des rapports, thèses, exemples et première section, voici les champs retenus. Tous sont spécifiés par du texte, par conséquent, définir des règles sur ce contenu n'a pas de sens.

Nom : c'est le nom donné par l'analyste à la tâche.

But : désigne le but assigné à la tâche.

Observations : remarques de l'utilisateur sur la tâche. Ce champ joue le rôle de bloc-notes et peut éventuellement contenir des médias dont l'importance est soulignée par [8].

Objets et Ressources utilisés : recense par une notation non-formelle les objets et les ressources utilisés dans l'accomplissement de la tâche. Les ressources regroupent toutes sortes de moyens employés par l'utilisateur (outils, documents papiers, etc.). Les objets sont ceux qui influencent l'activité. Mais comme déjà dit aucune distinction ne peut être réalisée à ce niveau.

Feed back : permet de décrire les effets observables (ou souhaitables) par l'utilisateur. Ce champ n'est accessible que pour les tâches dont l'exécutant fait intervenir le système. C'est une règle d'**existence conditionnelle**. Ce champ bien que textuel, peut servir comme point d'entrée pour une description plus finalisée de type UAN.

Durée : permet d'avoir une information sur la durée de la tâche. Le champ étant textuel, l'analyste est libre de rentrer toutes les informations disponibles qu'il a à ce sujet.

Numéro : est un numéro qui aide l'analyste à se localiser dans son arbre de description. Le numéro est créé à partir du numéro de sa tâche mère et de sa place parmi ses tâches sœurs.

Ces champs peuvent être perçus comme des remarques, mais orientés, contrairement au champ observation qui, à en juger par les documents, est peu utilisé. Pour les raisons exprimées lors de la première section les champs relatifs aux états (initial et final) sont supprimés.

Caractéristiques de la tâche.

Exécutant : indique qui réalise la tâche, c'est un énuméré {Utilisateur, Système, Interactif, Abstrait}. Ce champ comprend des **règles de constructions** : une tâche abstraite est composée de toutes sortes de tâches ; une tâche système est composée exclusivement de sous-tâches systèmes ; une tâche utilisateur est composée exclusivement de sous-tâches utilisateurs ; Une tâche interactive peut être composée de sous-tâches utilisateurs, interactives, systèmes mais en aucun cas abstraite ; une tâche abstraite ne peut pas être une feuille de l'arbre.

La notion de l'exécutant de la tâche est fondamentale. Elle trace la frontière entre ce qui est du ressort de l'utilisateur et ce qui est du ressort du système. Ce champ fixe par conséquent, la configuration de la tâche et permet d'obtenir des catégories, indispensables, pour envisager l'intégration de modules plus spécialisés.

Modalité : précise la forme particulière que revêt l'action de l'utilisateur. C'est un énuméré {sensori-motrice, cognitive}. Il y a une **existence conditionnelle** : l'exécutant doit être : « utilisateur » ou « interactif ». Une autre règle est que pour les tâches interactives la valeur est automatiquement égale à sensori-motrice.

Cette caractéristique est présente dans ce noyau du fait de l'importance que revêt la nature de la tâche utilisateur dans l'évaluation des tâches. Par l'intermédiaire d'un module plus spécialisé ce champ peut être affiné. Par exemple, l'évaluation de la charge de travail semble indiquer que la modalité pour les tâches cognitives de « résolution de problèmes » est à prendre en compte.

Fréquence : cette caractéristique tient lieu d'appréciation, du point de vue de l'utilisateur, du caractère répétitif de la tâche. C'est un énuméré {faible, moyenne, élevée} + un champ texte.

Ce noyau reprend la notion d'énuméré ainsi que les valeurs composant cette échelle d'évaluation. Cependant un échelon est ajouté, les exemples relevés dans la littérature montrent que des informations précises, sur la fréquence d'utilisation de la tâche, sont obtenues lors de l'analyse. Il convient d'en tenir compte ainsi, en plus de l'énuméré, un champ textuel est disponible. Ce champ, ne pouvant faire l'objet d'interprétation, il n'est disponible que si une valeur contenue dans l'énuméré est préalablement renseignée.

Importance : caractérise aux yeux de l'utilisateur l'intérêt de la tâche dans l'activité. C'est un énuméré {très importante, importante, peu importante}.

Bien que ce critère ne soit pas objectif, il permet par la suite de déterminer un ordre dans la présentation des tâches ou dans leurs exécutions. La fréquence et l'importance sont des attributs, certes utiles à l'évaluation du modèle, mais sont aussi exploitables dans le cadre d'un prototypage d'une maquette.

Evènements engendrés : c'est la liste des évènements pouvant être provoqués lors de l'exécution de la tâche. Par exemple la tâche « lancer impression » peut générer l'événement « plus de papier ».

Un événement engendré par une tâche est un fait qui ne résulte pas de la volonté de l'utilisateur, mais, qui peut influencer son activité. Cet événement est pris parmi ceux déjà décrits. Il doit être de source « provoquer ».

Ordonnancement local de la tâche.
Nécessité : indique si la tâche est facultative ou obligatoire. C'est un énuméré (de façon à avoir une valeur explicite). Le noyau dispose de cette caractéristique dans la mesure où l'ensemble des outils le prend en considération. Il y a une **existence conditionnelle** : une tâche réalisée par le système n'a pas cet attribut, de même avec l'opérateur « choix ».

Interruptible : indique que la tâche peut être interrompue. C'est un énuméré.

Itération : est une simplification d'écriture pour une tâche devant se répéter plusieurs fois ou devant être exécutée jusqu'à la satisfaction d'une condition. Elle est de type entier ou contient une condition (qui s'appuie sur une grammaire utilisant les objets de l'utilisateur).

Liste des utilisateurs : c'est la liste des utilisateurs pouvant exécuter la tâche. Elle est alimentée par les utilisateurs créés préalablement. L'association d'un utilisateur et d'une tâche engendre la description de deux autres caractéristiques : l'expérience et la compétence.

L'expérience est le niveau d'expertise de l'utilisateur dans la tâche réalisée. Cette expertise est propre à la tâche et à l'utilisateur. C'est un énuméré {novice, moyen, expert}. La compétence indique le type de compétences nécessaires à l'accomplissement de la tâche. C'est du texte.

La liste des utilisateurs est obligatoire pour les tâches interactives. Il doit y avoir au moins un utilisateur. Elle est facultative pour les tâches abstraites et non accessible pour les tâches systèmes. Il ne peut pas y avoir de doublon dans une même liste.

Précondition : c'est la condition obligatoire à respecter pour que cette tâche puisse être exécutable. Elle est soit textuelle soit formalisée par une grammaire qui met en relation la tâche et les objets de l'utilisateur.

Evènement déclencheur : cette condition indique que l'exécution est contrainte par un évènement. Par exemple la tâche « faire la procédure d'extinction des feux » n'a de sens que si l'événement « incendie » est survenu. La règle, pour les événements provoqués, est qu'une tâche ayant un évènement déclencheur peut contenir dans son sous-arbre une tâche qui engendre cet événement mais doit impérativement, pour l'autre sous-ensemble, avoir une tâche qui contienne cet évènement engendré.

Dans ce noyau, une tâche a soit une précondition, soit un événement déclencheur.

Manipulation des objets : ceci permet de donner une dynamique aux objets du modèle en modifiant les valeurs des attributs, en créant ou en supprimant des objets. Elle peut être textuelle dans le cas où le module des objets n'est pas utilisé, sinon il y a une grammaire.

Ordonnancement global de la tâche. Cette partie a trait aux opérateurs d'ordonnancements. Le lien est réalisé de la tâche mère à ses sous-tâches. Nous proposons une simplification en étudiant les dimensions régissant le séquencement des tâches et en laissant de côté les éventuels attributs (c.f. première section). Ces dimensions sont les suivantes :

- Les sous-tâches s'exécutent-elles toutes en même temps ;
- Plusieurs sous-tâches sont-elles exécutées ;
- Il y a t-il un ordre dans l'exécution.

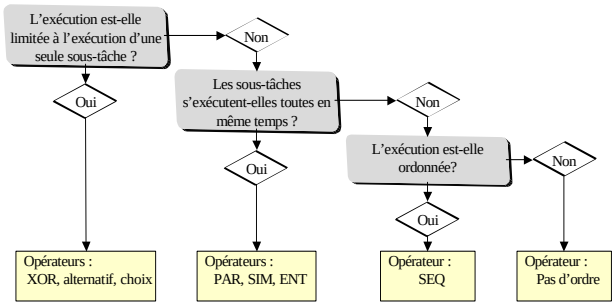

Figure 2 : Arbre de décision sur les dimensions intervenant dans l'ordonnancement et les opérateurs associés.

Cette approche présente trois avantages majeurs, le premier est que l'approche se base selon le point de vue de l'analyste, alors même que le modèle lui est adressé. Le second, est que l'on aboutit à une hiérarchie entre ces dimensions. La conséquence est qu'aux travers de ces dimensions nous pouvons trouver l'opérateur en adéquation par de simples questions. Enfin le troisième, est que l'on prouve que quatre opérateurs suffisent à exprimer l'ordonnancement entre la tâche mère et ses sous-tâches et cela sans faire d'impasse sur des opérateurs déjà présents. Néanmoins, dans ce noyau, les opérateurs sont aux nombres de sept :

- Alternatif : une seule sous-tâche est exécutée.
- Parallèle : les sous-tâches sont exécutées en même temps.

- Séquentiel : les sous-tâches sont exécutées une à une de gauche à droite.
- Pas d'ordre : les sous-tâches sont exécutées une à une dans n'importe quel ordre.
- Elémentaire : implique qu'une tâche n'est pas décomposable.
- Inconnu : permet de signaler que l'opérateur n'est pas connu.
- Alphanumérique : l'analyste peut spécifier lui-même l'opérateur. Dans ce cas, la simulation est impossible.

Il existe aujourd'hui un outil, écrit en java, qui permet de décrire l'activité conformément à la proposition du noyau. Les données sont gérées par un modèle sous-jacent qui est inspiré d'EXPRESS[3]. C'est un langage de description de données avec lequel nous pouvons exprimer des contraintes. La cohérence est contrôlée par l'application : des champs ne sont plus accessibles selon les valeurs des autres attributs, l'analyste est prévenu lorsque des liens entre des tâches ne respectent pas les règles, etc. La gestion des utilisateurs, des interruptions, des conditions, de la nécessité, de l'itération dans le cadre de la dynamique du modèle ont fortement participé à préciser la sémantique du modèle.

CONCLUSION

Nous avons présenté une classification suivant le pouvoir d'expression des modèles de tâches orientés utilisateur pour aboutir au fait que des passerelles ne peuvent pas être réalisées pour passer d'un outil à l'autre. Ceci nous a amené à une réflexion sur les usages de ces modèles. Ainsi une structuration par voie modulaire est proposée ainsi qu'un noyau contenant les données pour une description de l'activité. Cette description contient non seulement la notion d'événements, d'utilisateurs et de tâches, mais il permet aussi de lier formellement l'ensemble de ces notions. De plus, les opérateurs d'ordonnancements ont été simplifiés pour faciliter la description.

Cependant, bien que ce noyau ait été créé à partir des modèles existants et des descriptions d'activités déjà réalisées, il souffre d'un manque d'expérience. Ceci étant, il existe un outil –condition préalable à l'étude de terrain- qui va permettre de réaliser cette étude. L'outil intègre un module pour décrire les objets des utilisateurs. Ainsi il est possible d'écrire des préconditions, des post-conditions et des itérations qui tiennent compte de ces objets.

BIBLIOGRAPHIE

1. CTTE Version 1.5.3 téléchargée à l'adresse suivante http://giove.cnuce.cnr.it/tools2.html
2. Euterpe Version 1.00.91 à l'adresse suivante : http://www.cs.vu.nl/~martijn/gta/bin/Euterpe1.00.92.zip
3. ISO 10303-11, « *Product data representation and exchange : the EXPRESS language* », ISO TC184, Geneve, 1994
4. ISO, « *Information Processing Systems – Open Systems Interconnection – LOTOS – A Formal Description Based on Temporal Ordering of Observational Behaviour* », ISO/IS 8807. Central Secretariat, 1988
5. Jambon F, Brun P, Aït-Ameur Y; «*Spécification des systèmes interactifs*»; Analyse et Conception de l'IHM, Interaction homme-machine pour les SI1 ; dir. Kolski C; Hermès Science Publications; chapitre 6, pp 175-206, 2001
6. Johnson P., Johnson H., Waddington R., Shouls A. ; « *Task-Related Knowledge Structures: Analysis, Modelling and Application » ;* People and Computers IV, Proceedings 4th British Computer Society HCI group, 1988
7. Hix D., Hartson H.R., "Developping user interfaces: Ensuring usability through product & process". Newyork, USA: John Wiley & Sons, inc., 1993
8. Kaindl H ; « The Missing Link in Requirements Engineering » ; ACM SIGSOFT Software Engineering Notes ; vol 18 no 2 ; April ; pp. 3039 ; 1993
9. Limbourg Q., Vanderdonckt J. ; « *Comparing Task Models for User Interface Design* », dans The Handook of Task Analysis for Humain-Computer Interaction ; édité par Dan Diaper, Neville Stanton, Lawrence Erlbaum Associates Inc, 2004, pages 650, pages 135-154.
10. Paterno F., "Model-Based Design and Evaluation of Interactive Applications", Springer Verlag, 1999
11. Scapin D.L ; «*Analyse des tâches et aides ergonomique à la conception : approche MAD**»; Analyse et Conception de l'IHM, Interaction homme-machine pour les SI1 ; dir. Kolski C; Hermès Science Publications; chapitre 3, pp 85-116, 2001
12. Sebillotte S., Alonso B.M.., Fallah D., Hamouche H., Scapin D.L., Värnild E. ; *« Note de recherche concernant le formalisme MAD »*, INRIA Rocquencourt, Novembre 1994, 31 pages.
13. Tabary J.C, Barthet M.F ; «*Analyse et modélisation des tâches dans la conception des systèmes d'information : la méthode Diane+*»; Analyse et Conception de l'IHM, Interaction homme-machine pour les SI1 ; dir. Kolski C; Hermès Science Publications; chapitre 4, pp 117-144, 2001
14. Tamot 3 version 1.90 téléchargé à l'adresse suivante http://www.cmis.csiro.au/iit/Projects/Isolde/Tamot
15. van Welie M, van der Veer G.C, Eliëns A ; « *Euterpe - Tool support for analyzing cooperative environments » ;* Proceedings of the Ninth European Conference on Cognitive Ergonomics ; August 24-26 ; Limerick ; Ireland ; 1998