Une Approche Formelle pour l'Evaluation de la Tolérance aux Interruptions des Système Interactifs

Philippe Palanque, Jean-François Ladry, Eric Barboni, David Navarre, Marco Winckler

Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier
118 route de Narbonne
31062 Toulouse Cedex 9, France

RESUME
Ce papier présente une approche permettant d'étudier les éventuels effets des interruptions sur la performance d'exécution de la tâche dans un environnement multitâche. L'approche combine de précédents travaux dans le domaine de l'analyse d'interruption, des techniques de description formelles pour les systèmes interactifs et des processus stochastiques pour permettre l'analyse de performance de tâche utilisateur perturbée par des interruptions dans son environnement de travail. L'approche utilise des techniques de description formelle pour permettre une description complète des tâches utilisateurs, du système et du comportement des interruptions. Le mécanisme détaillé du comportement du système et des interruptions est présenté en utilisant un formalisme basé sur les réseaux de Petri appelé Interactive Cooperative Objects (ICO). L'utilisation de techniques de modélisation formelles pour ces trois composants permet de les comparer, de les analyser et de les intégrer. En particulier, cela nous permet de déterminer quels états du système sont affectés par les occurrences des interruptions. L'approche est illustrée par une étude de cas implémentant deux techniques d'interaction permettant la manipulation d'icônes dans un environnement de travail.

MOTS CLES : Approches basées modèle, techniques de descriptions formelles, interruptions, évaluation de performance.

ABSTRACT
This paper presents an approach for investigating potential disruptive effects of interruptions on task performance in a multitasking environment. The approach combines previous work in the field of interruption analysis, formal description techniques for interactive systems and stochastic processes to support performance analysis of user tasks constrained by the occurrence of interruptions in the working environment. The approach uses formal description techniques to provide a comprehensive description of user tasks, system and interruption behaviour. The detailed mechanism by which systems and interruptions behave is presented using a Petri nets-based formal description technique called Interactive Cooperative Objects (ICO). The use of a formal modeling technique for the description of these three components makes it possible to compare, analyze and integrate them. In particular, it allows us to determine which of the system states are actually affected by the occurrence of interruptions. The approach is exemplified by a case study that implements two interaction techniques for manipulating icons in a desktop environment.

IHM 2009, 13-16 Octobre 2009, Grenoble, France

CATEGORIES AND SUBJECT DESCRIPTORS: H5.m. Information interfaces and presentation (ex., HCI): Miscellaneous.

GENERAL TERMS: Documentation.

KEYWORDS: Model-Based approaches, formal description techniques, interruptions, performance evaluation.

INTRODUCTION

De nombreux environnements de travail sont multitâches et demandent à l'utilisateur de suspendre sa tâche courante afin de gérer une activité inattendue [30]. Dans des systèmes multitâches, les interruptions sont considérées comme de simples arrêts dans l'exécution de la tâche en cours, provocant des perturbations (attendues ou inattendues) dans le flux de contrôle. Cependant, des recherches dans le domaine de l'IHM ont montré qu'un changement efficace de tâches n'implique pas forcement que les utilisateurs seront capables de suspendre et reprendre une tâche de manière efficace [18]. Certaines études empiriques ont essayé d'élucider les effets des changement de tâches et leurs effets perturbateurs durant les interactions avec l'ordinateur [3][7][11][15][31]. Il a été démontré que les interruptions peuvent conduire à des incidents mettant en cause des erreurs humaines, comme par exemple, le cas de crashs d'avion liés à l'interruption de leurs pilotes pendant les phases de check-list d'avant vol [13]. La littérature actuelle sur les interruptions en interaction homme machine envisage le problème selon les perspectives suivantes: a) psychologie des interruptions humaine [31]; b) technologie pour améliorer la qualité de la génération d'interruption [23]; c) méthodes IHM pour la transmission des interruptions [19]; d) effets des interruptions dans le milieu de travail [11]; et e) études de cas décrivant les résultats de l'introduction de technologie dans les environnements de travail visant à améliorer les performances de coordination [24]. Le travail présenté

dans cet article introduit les premières briques servant de base à une nouvelle perspective pour la recherche dans ce domaine basée sur des techniques d'analyse de modèles afin d'étudier les éventuels effets perturbateurs d'une interruption sur l'exécution de tâches dans un environnement multitâche. L'approche combine de précédents travaux issus du domaine de l'analyse d'interruptions, des techniques de description formelles pour les systèmes interactifs et des méthodes d'analyse de performance pour comprendre les tâches utilisateur contraintes par les interruptions dans son environnement de travail. L'approche met en œuvre des techniques de description formelle pour permettre une description complète des tâches utilisateurs, du système et du comportement des interruptions.

ETAT DE L'ART

De nombreuses études empiriques essayent d'élucider les effets du changement de tâche et les effets perturbateurs qu'ils induisent sur l'activité de l'utilisateur durant une interaction avec un ordinateur (pour un examen complet de la littérature, voir MacFarlane & Latorella, 2002 [25]; Oulasvirta & Saarilouma, 2006 [31]; Trafton & Monk, 2006 [37]). Nous proposons certains des travaux les plus intéressants se reportant à notre étude.

La nature intrinsèque des interruptions

L'interruption de tâche sur un lieu de travail est un phénomène très répandu. Cependant pour mieux comprendre les effets des interruptions sur l'activité humaine il est important d'avoir une compréhension claire de ce qui amène à un changement de tâche. D'après Trafton et Monk [37], les interruptions surviennent lorsque qu'une personne travaille sur une tâche primaire (habituellement longue) et qu'une alerte survient concernant une tâche secondaire. Un aspect important des alertes réside dans le fait qu'elles sollicitent une attention de la part de l'utilisateur, introduisant ainsi la notion de temps nécessaire pour que l'utilisateur soit effectivement interrompu. La personne accomplit ensuite la tâche secondaire avant de reprendre la tâche principale. Pour retourner à la tâche principale la personne doit se souvenir à quel point elle en était dans cette tâche et ce qu'elle devait faire ensuite.

Les alertes peuvent être internes (*i.e.* pensées de l'utilisateur) ou externes (*i.e.* une alarme incendie, un message instantané). Les interruptions internes sont très difficiles à détecter et dans certains cas elles peuvent conduire à l'abandon de la tâche. Ceci peut être considéré comme une déviation normale du scenario d'utilisation, comme par exemple l'abandon du but initial par l'utilisateur. Les sources d'interruptions externes sont très variées allant d'évènements sociaux ou environnementaux à des alarmes systèmes. Cependant, si les interruptions sont levées dans un environnement informatique, des stratégies peuvent être employées pour décider quand on peut interrompre l'utilisateur pendant une interaction multitâche [24]. La compréhension du comportement de l'utilisateur dans chaque phase du cycle de vie de l'interruption amène des questions au niveau pratique et théorique. Iqbal et Horvitz [18] soutiennent qu'en fonction de la phase dans laquelle on se situe, différentes stratégies peuvent être employées pour l'utilisateur dans le changement de tâche en environnement multitâche.

Les interruptions et leurs facteurs perturbateurs

Différents types d'interruptions peuvent perturber les utilisateurs [4][10]. Souvent, les interruptions sont associées à des effets négatifs : a) continuer une tâche après une interruption est difficile et peut prendre du temps ; b) les tâches interrompues sont perçues comme plus dures que celles qui ne sont pas interrompues : c) les interruptions augmentent la charge cognitive et sont souvent frustrantes car elles empêchent les gens de finir leur travail ; d) des interruptions fréquentes peuvent réduire la performance de l'utilisateur.

Cependant toutes les interruptions ne sont pas négatives : certaines alarmes ou alertes systèmes peuvent déplacer l'attention de l'utilisateur vers un élément nécessitant une attention immédiate [23], et, pour des tâches simples et répétitives, les interruptions peuvent améliorer les performances [35]. Gillie et Broadbent [15] ont montré que d'être capable de répéter une étape dans la tâche principale ne protège pas des effets perturbateurs d'une interruption. Ils ont prouvé que des perturbations avec un contenu similaire peuvent être aussi perturbatrices même si elles sont très courtes. McFarlane [24] a examiné quatre politiques pour interrompre une personne durant une utilisation d'environnement multitâche (i.e. une réponse immédiate de l'utilisateur est requise ; l'utilisateur négocie quand il veut faire la tâche secondaire ; des agents intelligents déterminent quand interrompre l'utilisateur ; les interruptions arrivent à une fréquence pré arrangée). Il a mis en évidence que la performance est affectée par les politiques employées, mais il n'y a pas une politique meilleure que les autres. Par exemple, la réponse immédiate a montrée la plus mauvaise performance en termes de précision, mais la meilleure performance en termes de complétion. D'autres chercheurs ont étudié le timing des interruptions et de quelle manière une alerte peut permettre à une personne d'anticiper une interruption [13, 17]. Les alertes créent une latence correspondant au temps nécessaire pour la perception de l'interruption et les résultats de ces études ont montré que cette latence peut réduire les effets perturbateurs de l'interruption, principalement en réduisant le temps de réorientation sur la tâche primaire après que la tâche de l'interruption soit terminées mais aussi en en réduisant la le temps global de réalisation de la tâche primaire. Les délais d'interruption dans ces études permettaient aux participant de soit finir ce qu'ils étaient en train de faire avant de passer à l'interruption ou d'enregistrer des clés pour leur permettre de reprendre plus facilement la tâche primaire après avoir terminé l'interruption.

Survivre aux interruptions

Il existe plusieurs pistes sur la façon de construire un système interactif de façon à réduire les effets perturbateurs des interruptions. Par exemple, former les utilisateurs devrait réduire les effets perturbateurs des interruptions [37]. Cependant, apprendre à reprendre une tâche après une interruption implique que les interruptions et les reprises feront forcément partie de la formation. Trafton et Monk [37] ont montré que les interruptions deviennent moins perturbatrices avec le temps, l'expérience et la pratique du processus de reprise; tandis que l'expérience au niveau de la tâche primaire seule (sans interruptions) ne réduit aucunement les effets perturbateurs des interruptions. Cette étude recommande fortement que les formateurs dans des domaines complexes introduisent des scenarios de formation prenant en compte les interruptions occasionnelles pour réduire les perturbations dues aux interruptions lorsque les personnes seront sur leur lieu de travail.

Des recherches sur les processus de changement de tâche ont montré que les personnes avaient du mal à se rappeler de basculer entre les opérations. Afin d'améliorer la performance de l'utilisateur il a été suggéré l'utilisation d'indices signalant le moment où la tâche secondaire a besoin de l'attention de l'utilisateur [16]. Cela suggère que fournir une aide mnémotechnique externe peut grandement augmenter les performances pour les personnes ayant à gérer des interruptions critiques et des tâches cognitives éventuelles. En utilisant l'optique de la mémoire de travail à long terme (LTWM) Oulasvirta et Saariluoma [31] ont proposé de nombreuses suggestions pratiques. Sur la base des résultats de leurs expériences il a été suggéré que les concepteurs de systèmes devraient raccourcir la séquence des actions utilisateurs le plus possible. La durée de la séquence ne semble pas être théoriquement déterminée mais il semble que vingt secondes est une heuristique utilisée par certains concepteurs. Ils suggèrent aussi d'empêcher les interruptions lorsque la tâche requiert un temps important d'encodage (i.*e.* check-lists que doivent parcourir les pilotes).

Certains outils comme GroupBar développé à Microsoft Research [11] ont été conçus spécialement pour permettre aux personnes de sauvegarder et de retrouver leur configuration d'application et de gestion de fenêtres, ce qui peut être utile lors d'un changement de tâche. Iqbal et Bailey [19] ont aussi construit un certains nombres d'outils basés sur des travaux empiriques. Ils ont trouvé que le meilleur moment où l'on peut interrompre les personnes est entre les points d'arrêt entre les tâches. Ils ont créé un outil permettant de détecter les charges de travail et l'outil permettait l'interruption des utilisateurs pendant les périodes de faible charge de travail.

Modélisation de l'interruption de tâche

Il y a eu plusieurs tentatives pour formaliser des modèles cognitifs décrivant les impacts des interruptions sur le comportement humain [3]. Cependant, seule une petite partie a pris en compte des techniques de description formelles pour décrire l'occurrence d'interruptions dans les spécifications du système [20].

Dans les environnements multitâches, les interruptions doivent être vues simplement comme une interruption dans l'exécution de la tâche courante qui cause une déviation (attendue ou inattendue) dans le contrôle de flux. Ce problème est bien connu dans le domaine des systèmes opérationnels où les occurrences d'interruption pendant l'exécution de programmes en parallèle présentent de nombreuses similarités avec la gestion multitâche dans l'activité humaine. Des travaux précédents dans le domaine des systèmes opérationnels ont montré que la description systématique de toutes les déviations sur le système de contrôle de flux était à peu près impossible car elle conduirait à un nombre exponentiel et imprédictible d'états. La non-prédictibilité des interruptions conduit à l'utilisation de modèles déclaratifs pour décrire ce qui doit être accompli par l'utilisateur du système (quoi qu'il arrive) plutôt que de décrire les étapes permettant d'y arriver. Malgré tout, il existe certaines situations où les interruptions des tâches doivent être considérées comme faisant partie du but de l'utilisateur, comme par exemple annuler une impression. De nombreuses notations sont disponibles pour modéliser les tâches [12]. Une notation appartenant à cette catégorie est Concur Task Tree (CTT) [34] qui propose explicitement un moyen de décrire une interruption dans la tâche grâce à un opérateur *suspend/resume* (i.e. |>) comme présenté sur la *Figure 1*. Cependant, les opérations décrivant des interruptions sur un modèle de tâches doivent être utilisées que si l'interruption de tâche fait partie du but de l'utilisateur. De la même manière que CTT, West et Nagy [39] ont ajouté des structures théoriques à la notation GOMS permettant de surmonter ses limitations pour l'analyse des changements de tâches.

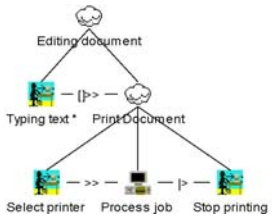

Figure 1. Exemple d'un modèle de tâche utilisant un opérateur d'interruption "|>" en CTT**.**

Dans une approche complètement différente Jambon [20] analyse les particularités des relations entre tâches (comme le *parallélisme, l'entrelacement et la séquence*) pour dériver un modèle formel (utilisant les automates) décrivant la sémantique des interruptions dans des notations comme MAD, UAN et les réseaux de Petri. Par exemple, si deux tâches s'exécutent en parallèle, il est admis qu'il n'y aura pas de perturbation durant leur exécution. Si deux tâches sont entrelacées, la perturbation équivaudra à l'effort à fournir pour changer de tâche. Si deux tâches sont supposées s'effectuer en séquence,

l'interruption d'une tâche pourra être interprétée comme une perturbation définie (*ex.* commencer la tâche *t2*, pourra interrompre l'éventuelle annulation de la tâche t1) ou un entrelacement entre les tâches avec une perturbation moindre sur l'activité humaine (*ex.* démarrer la tâche *t2* interrompra la tâche *t1* mais *t1* pourra être reprise lorsque la tâche *t2* sera terminée). De plus, la reprise de tâche peut être faite à n'importe quelle étape de l'exécution de tâche (*i.e.* redémarrage du début, reprise à l'étape précédant l'interruption et reprise à la fin de la tâche en supposant qu'elle a été accomplie).

L'APPROCHE

Cette section présente un processus de conception pour la conception et l'évaluation de systèmes tolérant aux interruptions. La *Figure 2* présente le processus itératif de construction des modèles utilisant des techniques de descriptions formelles. Au début du processus, un modèle préliminaire est construit puis analysé pour évaluer les propriétés qu'il vérifie. Ces propriétés peuvent être des propriétés de *vivance* comme "il arrivera éventuellement quelque chose de bien" ou des propriétés de *sureté* comme "il n'arrivera jamais quelque chose de mal" [22]. A partir de ce résultat d'analyse, on peut décider s'il y a lieu de modifier le modèle. Plus d'informations sur la façon d'entreprendre de telles vérifications dans le domaine des systèmes interactifs se trouvent dans [32].

Comme le but ultime d'un système critique est de permettre à l'opérateur d'accomplir son but de manière efficace et sans erreur, nous avons présenté un processus intégrant le modèle du système et le modèle de tâches. Le processus rend possible l'évaluation de conformité entre le modèle de tâches et le modèle du système [33]. Ce papier étend notre processus précédent pour comparer le niveau de tolérance aux interruptions des systèmes interactifs.

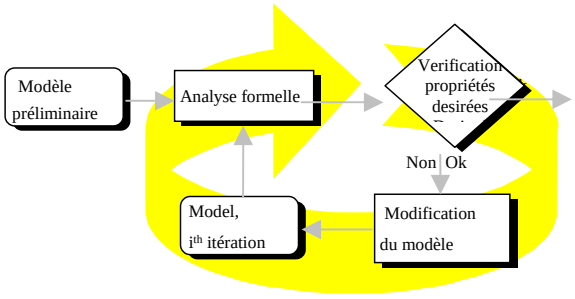

Figure 2. Processus itératif de construction du modèle

L'hypothèse de base, dans le processus de conception, est que les interruptions sont considérées comme des systèmes indépendants (appelé Source d'Interruption) situé dans l'environnement du système interactif à l'étude, et sont donc modélisées comme un système. Ici aussi, la modélisation commence par un modèle préliminaire qui pourra être modifié en fonction des résultats d'analyse ou suite à une analyse plus poussée des Sources d'Interruption. En effet, si nous laissons entendre qu'il n'y a qu'un modèle d'interruption, cela ne veut pas dire qu'il n'y a qu'un seul type d'interruption car plusieurs peuvent être intégrés dans un seul modèle. Il est important de noter que (la plupart du temps) le modèle des Sources d'Interruption ne fait que déclencher des éléments sur le modèle de tâche ou le modèle du système. Cependant, il peut y avoir des cas où le modèle des Sources d'Interruption doit être accédé depuis le modèle du système. Cela ne peut se faire que si une Source d'Interruption fournit une interface de programmation (API) pour intervenir sur le comportement prédéfini de la Source d'Interruption.

Activité de modélisation

La *Figure 3* présente le processus étendu montrant les modèles de tâches, du système et des interruptions. Un point critique du processus est l'activité de vérification de consistance (centre du diagramme). En effet, il est important de s'assurer que ces trois modèles présentent trois vues du même monde *i.e.* l'opérateur (modèle de tâches) interagit avec un système (modèle du système) pendant que des sources externes (modèle des Source d'Interruption) peuvent intégrer avec cette activité.

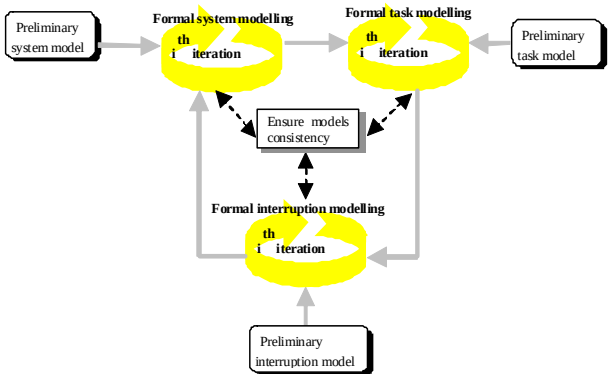

Figure 3. Processus itératifs impliquant des modèles de tâches, du système et des interruptions

Dans le domaine des systèmes critiques, les Sources d'Interruption peuvent être des éléments informatiques (comme des alarmes par exemple un avertissement de collision avec le sol (GCAS)) ou des éléments socio-technologiques (comme un contrôleur aérien demandant un changement de paramètre de vol – i.e. *clearance* - à un pilote). Dans un souci de brièveté et comme cela n'impacte pas l'approche que nous proposons, nous considérons que les interruptions font seulement partie du système externe et qu'elles seront donc modélisées indépendamment. Comme le but de l'approche est d'évaluer la tolérance aux interruptions, la *Figure 4* présente le processus permettant cette évaluation sur deux systèmes différents et de manière comparative. Les deux boîtes sur le côté gauche de la figure correspondent aux systèmes (système A en haut, système B en bas). Les deux systèmes étant destinés à supporter les mêmes buts et à recevoir les mêmes perturbations des Sources d'Interruptions, les modèles de tâches et d'interruptions sont les mêmes.

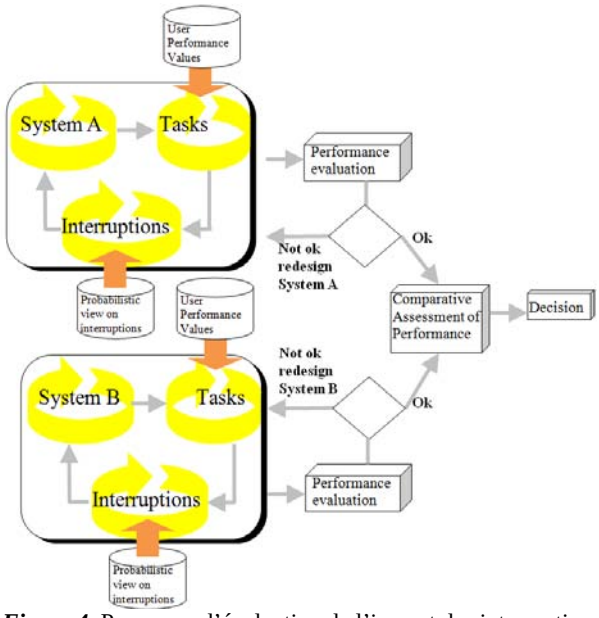

Figure 4. Processus d'évaluation de l'impact des interruptions sur deux systèmes différents

Injection de valeurs

Afin de pouvoir effectuer les évaluations de performances il faut pouvoir injecter des valeurs dans les modèles de tâches et d'interruption. Des valeurs de performances utilisateurs ont été ajoutées au modèle de tâche en fonction du type de tâche (motrice, cognitive, perceptive). Ces valeurs correspondent aux informations disponibles dans le domaine de l'IHM comme la loi de Fitts [14] pour les tâches motrices, dans le domaine des processeurs humains pour les valeurs concernant les performances cognitives et perceptives [6]. D'autres données peuvent être utilisées car ce domaine de recherche évolue régulièrement et des recherches fournissent des données sur la prédiction au niveau des techniques d'interaction, comme les mouvements contraints [1] et des interfaces zoomables [8]. La validité de ces données est critique lorsqu'il faut évaluer la performance du triplet (système, tâche, interruption). Cette phase est présentée dans la boite nommée "performance evaluation " de la *Figure 4*. Les résultats de cette phase sont utilisés comme fil conducteur pour décider (dans le cas de performances inférieures à celles attendues ou requises) de modifier des éléments du système (en changeant par exemple les techniques d'interaction) ou des tâches (en modifiant la formation des opérateurs et par conséquence les tâches qu'ils devront effectuer) ou enfin des interruptions (s'il est possible d'influencer certains événements externes). Cependant, il faut noter que les valeurs brutes ne sont pas très importantes si on se place au niveau d'une évaluation comparative. En effet, comme cette évaluation est relative, *i.e.* elle ne produit aucune mesure absolue de la tolérance aux interruptions mais une comparaison de deux systèmes (voir la boite "comparative assessment of performance" sur la *Figure 4*). Par conséquent, et comme les tâches en considération sont les mêmes, les valeurs n'influence pas les résultats de l'évaluation.

Instanciation de l'approche

La *Figure 5* présente l'instanciation du processus de la *Figure 4* incluant les notations et les valeurs présentées ci-dessus. Les cercles gris clair contiennent les théories à partir desquelles les valeurs sont obtenues ainsi que les notations permettant la construction des modèles. A noter que : a) les modèle du système et celui des interruptions sont décrits en utilisant la notation ICO ; et b) le modèle de tâche est décrit en utilisant la notation CTT.

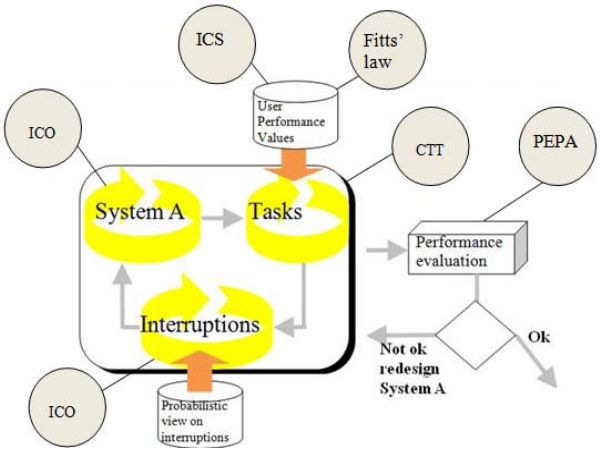

Figure 5. Instanciation du processus de la Figure 4

A ce stade le processus reste théorique et il reste possible de le rendre applicable à un large ensemble de systèmes interactifs dans divers domaines d'application. La section suivante présente la façon dont cette approche peut être appliquée sur une étude de cas qui présente les modèles, les transformations et les aspects évaluation de performance.

ETUDE DE CAS

Afin d'illustrer l'approche décrite dans la section précédente nous allons introduire une étude de cas simple. L'objectif de l'étude de cas est de présenter les différentes phases de l'approche sur une application réaliste mais simple. La *Figure 6* présente l'interface utilisateur de l'étude de cas. Dans cette application, un ensemble d'icônes est présenté sur une grille dans une fenêtre. Les icônes peuvent être déplacées ou effacées.

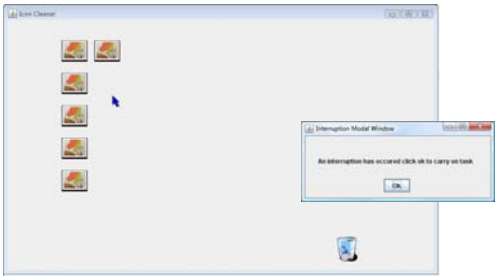

Figure 6. Interface utilisateur de l'étude de cas

La tâche de l'utilisateur est limitée à celle présentée dans la *Figure 7*. Le but de l'utilisateur (tâche du haut dans le modèle de tâches) est d'effacer tous les icônes de l'interface utilisateur. Ceci peut être réalisé en effectuant dans n'importe quel ordre la sélection d'un icône et

l'envoi d'une commande d'effacement. Pour atteindre son but, l'utilisateur doit effectuer de façon itérative une sélection puis un effacement (représenté dans le modèle par le symbole * dans la tâche abstraite "Clear Icons").

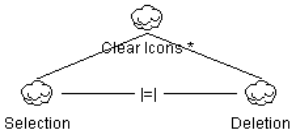

Figure 7. Modèle de tâche abstrait en CTT

Deux systèmes permettant de réaliser cette tâche ont été construits. Le premier : système A (appelé Drag & Drop) fournit une technique d'interaction de type Drag and Drop. Les icônes sur l'interface utilisateur peuvent être sélectionnés en bougeant le pointeur de la souris au dessus de l'icône et en appuyant sur le bouton gauche. Une fois sélectionnés, les icônes peuvent être déplacés dans la fenêtre à n'importe quel endroit. Si le bouton gauche est relâché quand la souris survole la corbeille alors l'icône est effacé. Le second : système B (appelé Speak & Click) fournit une technique d'interaction impliquant de la reconnaissance de parole (pour la commande d'effacement) et de geste (pour la sélection d'icône). Les modèles du système et les modèles de tâches relatifs à ces deux systèmes seront présentés dans les sous-sections suivantes.

Modélisation relative au système A (Drag & Drop)

Le comportement du système A a été complètement modélisé en utilisant la notation ICO. Il est présenté dans la *Figure 8*. Le modèle contient toutes les préconditions concernant l'état de l'interaction ainsi que l'ensemble des séquences d'événements qui sont disponibles à l'utilisateur.

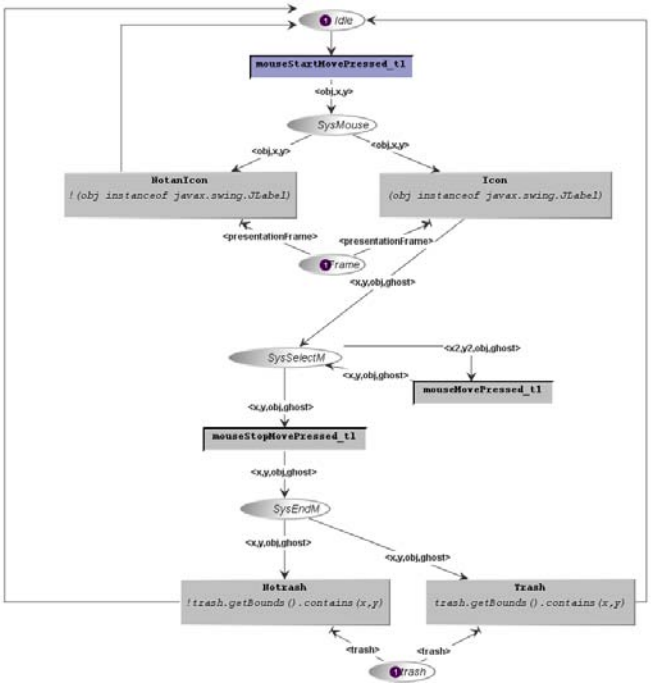

Figure 8. Modèle ICO du système A (Drag & Drop)

Comme on peut le voir dans la *Figure 3*, le système a un impact sur la manière dont les tâches vont être réalisées par l'utilisateur ; de ce fait le modèle de tâche doit être retravaillé pour être compatible avec le modèle du système. Le modèle de la *Figure 7* devient celui présenté par la *Figure 9*. Les tâches de sélection et d'effacement peuvent être raffinées. La sélection est réalisée par une succession de tâches (choix de l'icône à effacer, déplacement de la souris sur l'icône choisi, et par l'appui du bouton gauche de la souris). L'effacement est réalisé par une succession de tâches (déplacement de l'icône au dessus de la corbeille, vérification que l'icône de la corbeille est en surbrillance, et par le relâchement du bouton de la souris).

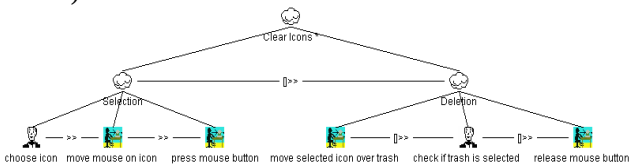

Figure 9. Modèle de tâche raffiné pour être conforme au comportement de Drag & Drop (système A)

Il est intéressant de noter que l'operateur temporel "order independence" entre les tâches de sélection et d'effacement est plus contraint dans la version raffinée que dans la version d'origine, remplacé par l'opérateur "sequence". Il est possible d'augmenter les contraintes sur un modèle si celui-ci reste compatible avec le modèle abstrait. L'inverse n'est pas possible car cela autoriserait l'utilisateur à effectuer des séquences d'action qui ne seraient pas autorisées dans le modèle abstrait.

Modélisation relative au système B (Speak & Click)

Comme pour le système A, le système B a été complètement décrit en utilisant le formalisme ICO (*Figure 10*).

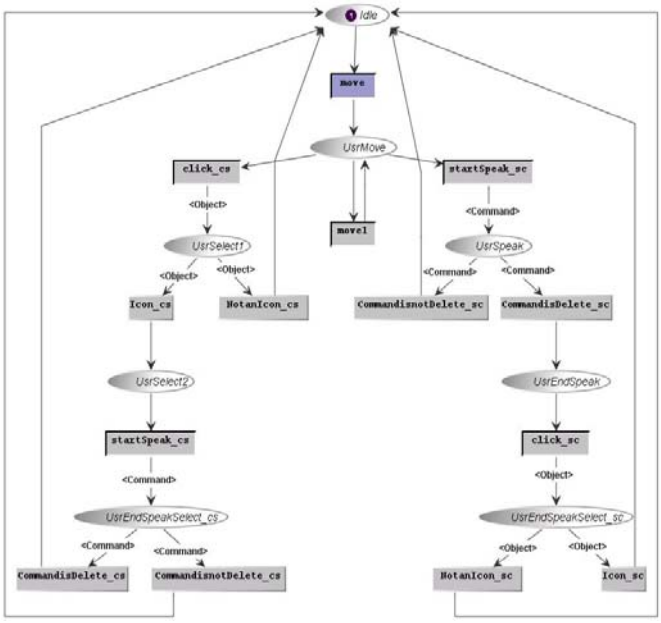

Figure 10. Modèle ICO du système B (Speak & Click)

Ce modèle autorise l'utilisateur soit à commencer par une commande vocale d'effacement "delete" puis à sélectionner un icône soit à commencer par sélectionner un icône à effacer puis à l'effacer en utilisant la commande d'effacement "delete". Tous les cas possibles sont représentés explicitement sur le modèle, par exemple, la commande "delete" non reconnue, la commande "delete" reconnue mais aucun icône sélectionné,… Le côté gauche du modèle correspond à la séquence où la sélection est effectuée en premier (les transitions sont labélisée "cs" Click and Speak) tandis que pour le coté droit, on commence par la commande vocale (on utilise alors "sc" pour Speak and Click). Dans tous les cas, l'interaction doit commencer par un mouvement de souris provoqué par l'utilisateur (ceci est représenté sur le modèle par les transitions du haut "*move*").

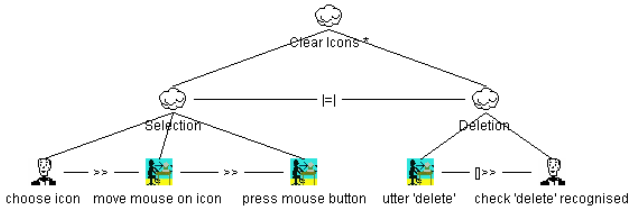

Figure 11. Modèle de tâche raffiné pour être conforme au comportement de Speak & Click (système B)

Comme pour le système A, le modèle de tâches du système B a été raffiné pour être compatible avec le modèle du système B. La *Figure 11* montre le modèle de tâches raffiné. Il est intéressant de noter que le modèle de tâches est moins contraint que pour le système A car il autorise les tâches de sélection et d'effacement dans n'importe quel ordre (opérateur "order independence" (|=|) au premier niveau du modèle de tâches).

Modélisation des interruptions

D'après l'approche décrite dans la *Figure 3* le dernier modèle à réaliser est le modèle décrivant le comportement des Sources d'Interruptions. Bien que l'état de l'art décrive précisément la nature des interruptions ainsi que de nombreux résultats de recherches, nous présentons ici le modèle d'interruption le plus simple possible. En effet, ce papier s'intéresse à la description d'un processus permettant l'intégration de modèles et comment ces modèles peuvent être intégrés pour permettre une évaluation de performance. Nous appliquons actuellement ce processus dans le domaine des cockpits interactifs, en étendant nos travaux précédents concernant la reconfiguration des cockpits interactifs d'avions en cas de panne matérielle des éléments d'affichages [27]. En effet, ces reconfigurations et ces erreurs sont perçues par les opérateurs comme des interruptions et les performances dans ce contexte sont un élément critique.

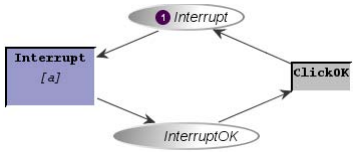

Figure 12. Modèle ICO d'une Source d'Interruption

L'interruption prise en compte se comporte comme suit : à chaque fois qu'une interruption survient, une fenêtre modale est affichée proposant un bouton Ok (fenêtre au centre de la *Figure 6*) et l'utilisateur doit cliquer sur le bouton pour pouvoir reprendre sa tâche initiale. La *Figure 12* présente ce comportement en utilisant la notation ICO. Dans l'état initial, l'état de l'interruption est au repos (un jeton *Interrupt*) dans l'attente de l'arrivée d'une interruption. Ceci est modélisé par la transition *Interrupt* qui est franchie en fonction d'une variable temporelle (représentée par le symbole *[a]*). Quand l'interruption est survenue, le jeton de la place *Interrupt* est déplacé sur la place *InterruptOk* et la transition *ClickOk* devient disponible. Cette transition est connectée au bouton Ok *i.e.* lorsque l'utilisateur clique sur le bouton un événement est envoyé au modèle. Ceci provoque le franchissement de la transition *ClickOk* et le modèle revient sans son état initial.

Compatibilité avec modèles de tâches

La dernière étape du processus est l'intégration du modèle de tâche avec les modèles résultant de la fusion des modèles du système et des modèles d'interruptions. Il faut noter que la construction du modèle global est rendue possible grâce à l'activité de vérification de consistance montrée dans la *Figure 3*, qui force l'itération jusqu'à obtention de la compatibilité entre modèles.

A l'image des travaux que nous avons déjà présentés dans [26, 28] il est possible de mettre en correspondance les modèles du système interactif et des tâches utilisateurs. De par la simplicité de cette étude de cas nous ne présentons pas plus avant l'exploitation de ce lien pour évaluer la compatibilité, mais le lecteur intéressé pourra trouver plus d'information dans [33].

Injection de valeurs

La dernière activité à effectuer avant de s'occuper de l'évaluation de performance est l'injection de valeur dans nos modèles pour représenter le moyen temps d'exécution d'une tâche accomplie par un utilisateur sur le système interactif modélisé. Ces valeurs peuvent être obtenues de trois sources différentes : observations empiriques (lorsque des résultats expérimentaux sont disponibles), données moyennes provenant de modèles théoriques (comme KLM, loi de Fitts, …) ou un ensemble de valeurs utilisées pour tester la validité d'une hypothèse comme nous l'avons vu plus haut pour la fréquence d'une interruption. Cette donnée est utilisée pour raffiner les valeurs génériques du modèle de processeur humain [9], de la loi de Fitts et des valeurs calculées sur des performances connues du système utilisé (par exemple, le système de reconnaissance vocale). Le *Tableau 1* correspond aux valeurs injectées dans le modèle qui décrit la technique d'interaction Speak&Click (voir *Figure 10*) à partir des tâches associées identifiées dans le modèle de la *Figure 11*. La première colonne correspond aux noms des transitions du modèle ICO, la deuxième colonne correspond au temps moyen estimés pour l'exécution de la

transition correspondante et la troisième colonne correspond à la source théorique des valeurs injectées.

Tableau 1. Valeurs injectées dans notre étude de cas**.**

Tâches / Transitions	**Temps** (/ms)	**Explication de la valeur**
'move mouse on icon" ➔ move, move1	190 + 100	**Fitts** pour une distance moyenne de 1200 pixels et une taille d'icône de 32 pixels **(+) Processeur humain** pour le temps cognitif de *choose icon*
'press mouse button" ➔ click_sc, click_cs	200	KLM, Keystroke model
'utter 'delete' ", "check 'delete' recognized" ➔ startSpeak_cs, startSpeak_sc	630 + 100	Prononciation du mot 'delete' + temps de reconnaissance vocale + temps cognitif (**processeur humain**) pour la tâche *check-delete-recognized*
Tâches systèmes ➔ Icon_cs, NotIcon_cs, Icon_sc, NotIcon_sc, CommandIsNotDelete_cs, CommandIsNotDelete_sc	2	Temps moyen d'évaluation des conditions de ces transitions

Le *Tableau 2* présente les valeurs à injecter dans le modèle d'interruption. Le lecteur attentif notera deux types de valeurs injectées :

- La durée de l'interruption elle-même calculée à partir des temps cognitifs (perception de l'interruption et reprise de la tâche) et du temps d'exécution de la tâche d'interruption (ici, cliquer sur le bouton OK).
- La fréquence de cette interruption est représentée par une valeur symbolique (*a*) que nous faisons varier dans le but de déterminer la performance du système pour différentes fréquences d'interruptions.

Tableau 2. Valeurs injectées dans le modèle d'interruption**.**

Transition	**Temps** (/ms)	**Explication de la valeur**
clickOK	100 + 200 + 100	**Processeur humain** pour le temps cognitif de perception de l'interruption **(+) KLM, Keystroke model** pour le clic sur le bouton OK **(+) Processeur humain** pour le temps de reprise de la tâche interrompue
interrupt	*a*	Paramètre de simulation de la fréquence

Pour injecter ces valeurs, nous produisons un réseau de Petri issu de la transformation des modèles ICO dans lesquels toutes les transitions sont remplacées par des transitions temporisées dont le temps d'exécution est donné par les valeurs du *Tableau 1*. Le réseau de Petri ainsi obtenu donne une représentation de l'utilisation du système modélisé par l'utilisateur représenté par les valeurs empiriques selon les modèles de tâches associés. Dans cet article nous nous limitons à des données empiriques issues de la littérature dont le seul but est d'exemplifier l'injection des valeurs. Ces valeurs pourrait être remplacées par des données réelles issues d'expérimentations prenant en compte les contraintes environnementales (matériel, bruit, …). Une façon de collecter de telles données pourraient être l'utilisation des modèles ICO eux-mêmes car ils sont exécutables dans leur environnement dédié PetShop [29] offrant ainsi un prototype de l'application elle-même. Dans [5] notamment, l'exploitation du mécanisme de log de PetShop permet de conserver une trace de l'exécution des modèles ICO de l'application, dans le but d'assister l'évaluation utilisateur, et dont une des retombées pourraient être l'identification de valeurs temporelles associable aux techniques d'interaction.

Evaluation de performance

Dans de précédents travaux nous nous sommes intéressés à exploiter les informations temporelles pour une analyse quantitative de la performance de plusieurs applications [21] dans le but d'assister la prise de décision dans un processus de conception. L'originalité du travail présenté ici réside dans la volonté d'évaluer la performance d'un système interactif face à l'occurrence d'interruption, c'est-à-dire la dégradation de l'interaction dans ce système interactif en fonction de la fréquence d'interruption. En se basant sur des valeurs estimées proches des temps moyens observés, une évaluation prédictive favoriserait :

- un meilleur ciblage des scénarios de test utilisateur (quelles étapes d'utilisation du système modélisé qui sont le plus affectées par des interruptions, …),
- la comparaison de plusieurs alternatives de conception (laquelle est la plus performante pour un sous-ensemble critique de tâches ou laquelle est la plus tolérante aux interruptions face aux mêmes tâches),
- le paramétrage à bas niveaux des techniques d'interaction modélisées (quelle performance minimale de la technique est acceptable pour que l'utilisateur puisse accomplir les tâches critiques).

En effet, comme il est discuté dans l'état de l'art de ce papier, on constate que l'impact de l'interruption sur l'utilisation du système présente deux niveaux [23] ; soit l'interruption reste une perturbation dans l'activité de l'utilisateur, soit cette dernière devient gênant au point de nuire à l'exécution de cette activité. Être capable d'estimer le moment où, pour un système donné, le niveau d'impact change permettrait d'influer :

- sur la conception du système lui-même en adaptant celui-ci au contexte d'interruption,
- sur le choix des scénarios de tests, en identifiant le type d'interruptions à inclure dans ces tests.

La base de la mise en œuvre de l'évaluation prédictive réside dans le modèle d'interruption pris comme paramètre d'expérimentation. Par exemple la valeur symbolique *a* du modèle présenté sur la *Figure 12* est le paramètre représentant la fréquence des interruptions. Dans [36], nous avons simulé le nombre de déposés effectifs dans la corbeille par l'utilisateur pendant une période fixée, en utilisant un premier type d'approche basé sur la transformation du réseau injecté en automate et en utilisant des outils d'analyse de performance possédant un langage déclaratifs pour l'expression des scénarios d'évaluation (c'est par exemple le couple PEPA et PRISM) La *Figure 13* donne une représentation du nom-

bre de déposés pour les deux systèmes en fonction de la fréquence d'interruption.

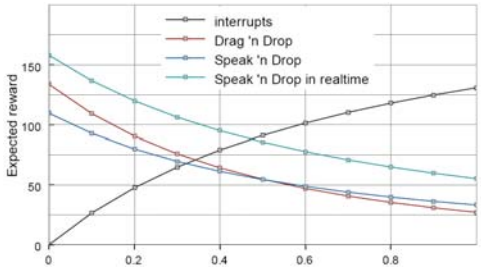

Figure 13. Comparison of performance of Drag & Drop, Speak & Drop, and Speak & Drop with realtime speech recognition

Par exemple, la *Figure 13* montre qu'avec un taux faible d'interruptions, la technique Speak and Drop est moins efficace que le classique Drag and Drop, et que la tendance s'inverse dès le franchissement d'un certain seuil.

Cette approche, bien que prometteuse présente l'inconvénient de pertes potentielles d'information au cours de la transformation en automate de par le pouvoir d'expression plus limité du dialecte utilisé. D'autres approches existantes nous ont permis d'identifier deux autres voies prometteuses :

- Utilisation de dialecte de réseaux de Petri dédié à l'évaluation de performance, ce qui sous-entend une transformation du réseau de Petri injecté vers le dialecte cible. Les Generalized Stochastic Petri Nets (GSPN) [2] sont un exemple d'un tel dialecte.
- Instrumentation de PetShop ou exploitation du mécanisme de log existant, mais cette voie n'a pas été explorée pour le moment.

Cette brique du processus que nous présentons dans ce papier est toutefois toujours en cours de réflexion.

DISCUSSION ET CONCLUSION

Les interruptions sont imprévisibles et ne peuvent généralement pas être ignorés par l'utilisateur. Comme les utilisateurs sont confrontés à des sources de plus en plus nombreuses d'informations qui nécessitent leur attention il est important de connaître la façon dont ces interruptions affectent leur capacité à achever leurs tâches. Dans le contexte de systèmes interactifs critiques il faut être capable de prévoir l'impact des interruptions durant l'exécution d'une tâche.

Dans ce papier nous avons proposé un processus de conception pour l'évaluation de systèmes tolérants aux interruptions. La faisabilité d'une telle approche est illustrée par une étude de cas qui nous permet de comparer deux techniques d'interactions en ce qui concerne leur tolérance aux interruptions. Cette étude de cas se veut minimaliste pour mettre en évidence l'approche plutôt que le système en lui même. Le travail présenté ici met en avant les aspects fréquence d'interruption dans l'analyse de performance, mais l'approche est suffisamment générique pour permettre d'exprimer d'autres dimensions de l'interruption telles que le type (physique ou cognitif), la durée ou le nombre (plusieurs interruptions concurrentes). Même si l'analyse de performance est une technique connue pour l'évaluation de systèmes multitâches [38], nous n'avons pu trouver dans la littérature l'évaluation de l'effet des interruptions sur des tâches utilisateur.

En injectant des valeurs nous pouvons évaluer les performances des paramètres que nous étudions. Cependant, la précision des valeurs injectées pour l'analyse de performance n'est pas le but principal de cet article. L'objectif principal est de fournir un outil pour évaluer l'impact des interruptions sur les performances des tâches plutôt que des informations précises sur l'exécution de l'application. De plus, l'analyse de performance devrait être vue au niveau plus large de l'expérience utilisateur et le résultat devrait être considéré de paire avec l'utilisabilité perçue du système. C'est pourquoi, ce travail est à considéré comme une première tentative de méthode de conception globale pour des systèmes interactifs tolérants aux interruptions.

BIBLIOGRAPHIE

1. Accot J. & Zhai S. Beyond Fitts' law: models for trajectory-based HCI tasks. In Proc. of ACM CHI'97, 1997, pp. 295–302
2. Ajmone Marsan, M.; Balbo, G.; Conte, C.; Donatelli, Susanna, & Franceschinis, G. Modelling with generalized stochastic Petri nets. Wiley; 1995.
3. Altmann, E. M., Trafton, J. G. Timecourse of Recovery from Task Interruption: Data and a Model. Psychonomics Bulletin and Review 14(6).
4. Bailey, B.P. Konstan, J.A. & Carlis, J.V. Measuring the effects of interruptions on task performance in the user interface. In Proc. of the IEEE International Conference on Systems, Man, and Cybernetics 2000. Nashville, USA. Vol. 2, pp. 757-762.
5. Bernhaupt R., Navarre D., Palanque P. & Winckler M. Model-Based Evaluation: A New Way to Support Usability Evaluation of Multimodal Interactive Applications. Maturing Usability: Quality in Software, Interaction and Value, Springer HCI series.
6. Bourgeois, F., Guiard, Y., & Beaudouin-Lafon, M. Multi-scale pointing: Facilitating pan-zoom coordination. In Proc. ACM CHI'02, pp. 758-759.
7. Cades D. M., Trafton J. G., Boehm-Davis D. A. & Monk C. A. Does the difficulty of an interruption affect our ability to resume? In Proc. of the 51st HFES Conf., Santa Monica, USA, pp. 234-238
8. Card, S.K.; T.P. Thomas & A. Newell, written at London, The Psychology of Human-Computer Interaction, Lawrence Erbaum Associates.
9. Card, S K.; Moran, Thomas P., & Newell, Allen. The Model Human Processor: An Engineering Model of Human Performance. Handbook of Perception and Human Performance; 1986: pp. 1-35.

10. Czerwinski, M., Cutrell, E. & Horvitz, E. Instant Messaging and Interruption: Influence of Task Type on Performance, OZCHI'2000, Sydney, Australia.
11. Czerwinski, M., Horvitz, E., & Wilhite, S. A diary study of task switching and interruptions. In Proc. ACM CHI'04, Vienna, Austria, 2004, pp. 175-182.
12. Diaper, D. & Stanton, N. A. (eds.) The Handbook of Task Analysis for Human-Computer Interaction. Lawrence Erlbaum Associates, 2004. 650 pgs
13. Diez M., Boehm-Davis D. A. & Holt R. W. Model-based predictions of interrupted checklists, In Proc. of the 46th HFES Conf. Santa Monica, USA.
14. Fitts, P. M.: The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. J. of Experimental Psychology, N. 47.
15. Gillie T. & Broadbent D. What makes interruptions disruptive? A study of length, similarity and complexity, Psychological Research, 50 (4).
16. Hopp, P.J, Smith, C. A. P. & Clegg, B. A., Heggestad, E.D. Interruption Management: The Use of Attention-Directing Tactile Cues. Human Factors, Vol. 47, No.1, 2005, pp.1–11.
17. Horvitz, E. & Apacible, J. 2003. Learning and reasoning about interruption. In Proc. of ACM ICMI'03, Vancouver, Canada, 2003, pp. 20-27.
18. Iqbal, S. T. & Horvitz, E. Disruption and Recovery of Computing Tasks: Field Study, Analysis, and Directions. In ACM CHI'07. San Jose, USA, 2007.
19. Iqbal, S. T. & Bailey, B. P. 2008. Effects of intelligent notification management on users and their tasks. In Proc. of ACM CHI '08. pp. 93-102.
20. Jambon F. Erreurs et interruptions du point de vue de l'ingénierie de l'interaction homme-machine. Thèse de Doctorat, Univ. Joseph Fourier, 1996.
21. Lacaze, X., Philippe, P., Navarre, D. & Bastide, R. Performance Evaluation as a Tool for Quantitative Assessment of Complexity of Interactive Systems. In Proc. DSV-IS'02, University of Rostock, 2002.
22. Lamport L. proving correctness of multiprocess programs. IEEE TSE, Vol 3 n°2, 125-143, 1977.
23. McCrickard D. S. & Chewar C. M. Attuning notification design to user goals and attention costs, Communications of ACM, 46 (3), 67-72.
24. McFarlane D. C. Coordinating the interruption of people in human-computer interaction. In INTERACT'99, Amsterdam, The Netherlands: IOS Press.
25. McFarlane D. C., Latorella K. A. The scope and importance of human interruption in human-computer interaction design. Human-Computer Interaction 17.
26. Navarre, D., Palanque, P., Barboni, E., Mistrzyk, T. On the Benefit of Synergistic Model-based Approach for Safety Critical Interactive System Testing. TAsk MOdels DIAgrams for UI design (TAMODIA'07), Toulouse, France, Springer-Verlag.
27. Navarre, D., Palanque, P. & Basnyat, S., (2008) Usability Service Continuation through Reconfiguration of Input and Output Devices in Safety Critical Interactive Systems. In Proc. of the SAFECOMP'08, Springer LNCS 5219, pp. 373–386.
28. Navarre, D., Palanque, P., Bastide, R., Paternò, F., Santoro, C. A Tool Suite for Integrating Task and System Models Through Scenarios. DSV-IS'2001, Glasgow, Scotland, Springer, lncs 2220, june 2001.
29. Navarre, D., Palanque, P., Bastide, R. & Sy, O. (2001). A Model-Based Tool for Interactive Prototyping of Highly Interactive Applications. in 12th IEEE, International Workshop on Rapid System Prototyping, Monterey (USA), IEEE Press 2001
30. O'Conaill B. & Frohlich D. (1995) Timespace in the workplace: Dealing with interruptions, in: Human Factors in Computing Systems: CHI'95 Companion, New York: ACM Press, 262-263
31. Oulasvirta, A. & Saariluoma, P. 2006. Surviving task interruptions: Investigating the implications of long-term working memory theory. Journal of Human Computer Studies. 64, 10 (Oct. 2006), 941-961.
32. Palanque P. & Bastide R. Verification of an Interactive Software by analysis of its formal specification. In Proc. of IFITP TC13 INTERACT'95, Lillehammer, Norway, 27-29 June 1995, p. 191-197.
33. Palanque, P., Bastide, R. Synergistic modelling of tasks, system and users using formal specification techniques. Interacting With Computers, Academic Press, 9, 12, pp. 129-153
34. Paterno, F., Mancini, C. & Meniconi, S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In: Proc. of INTERACT'97. Chapman & Hall (1997), 362-369.
35. Speier C., Vessey I. & Valacich J. S. (2003) The effects of interruptions, task complexity, and information presentation on computer-supported decision-making performance, Decision Sciences, 34 (4).
36. ter Beek M. H., Faconti G. P., Massink M., Palanque P., Winckler M. Resilience of Interaction Techniques to Interrupts: A Formal Model-based Approach. INTERACT'09. Uppsala, Sweeden.
37. Trafton, J. G., & Monk, C. A. (2007). Task Interruptions. Reviews of Human Factors and Ergonomics 3.
38. Waszniowski, L., Hanzálek, Z. (2008) Formal verification of multitasking applications based on timed automata model. Real-Time Syst. Vol. 38: 39–65.
39. West, R. L., Nagy, G. (2007) Using GOMS for Modeling Routine Tasks Within Complex Sociotechnical Systems: Connecting Macrocognitive Models to Microcognition. J. of Cog. Engineering and Decision Making, Vol. 1, n° 2, pp. 186-211.