Une approche à base de modèles pour l'ingénierie logicielle de techniques d'interaction

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

Une approche à base de modèles pour l'ingénierie logicielle de techniques d'interaction

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

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

RESUME : Alors que les approches à base de modèle sont utilisées depuis plus de 30 ans dans le domaine de la description comportementale de systèmes interactifs [27], le lien entre de telles approches et les processus de conception centrés utilisateur demeurent insuffisamment explicités. Cet article propose une contribution à cette problématique en présentant comment une approche à base de modèles peut être exploitée pour faciliter les tâches d'évaluation de l'utilisabilité qui sont souvent fastidieuses et répétitives. Le principe de base de cette approche favorise l'utilisation de l'enregistrement et l'analyse de données de log dans un environnement à base de modèles. Les résultats décrits dans cet article démontrent que les données de log au niveau des modèles peuvent être non seulement utilisées pour identifier des problèmes d'utilisabilité mais aussi pour identifier les modifications à apporter à ces modèles dans le but de corriger les problèmes rencontrés. Cette approche est intégrée dans un processus et est supportée par un environnement outillé basé sur les modèles permettant la modélisation, la simulation et l'évaluation des systèmes interactifs. L'étude de cas présentée illustre les principes de l'approche et le fonctionnement de l'outil sur une technique d'interaction. Elle montre comment l'analyse des données de log permet au concepteur de régler facilement la technique d'interaction (comme les résultats de l'analyse des données de log sont présentés au même niveau d'abstraction que les modèles). Elle propose une alternative aux tests utilisateur qui sont très difficile à configurer et à interpréter en particulier lorsque l'on considère des interfaces avec des techniques d'interaction avancées.

MOTS CLES : modélisation de systèmes interactifs, techniques d'interaction, évaluation de performances, évaluation de l'utilisabilité basée sur des modèles.

ABSTRACT

While model-based approaches have been used for over 30 years in the field of behavioral description of interactive systems [27], the link between these approaches and user-centered design process remain insufficiently explained. This paper offers a contribution to this problem by presenting how a model-based approach can be exploited to facilitate the tasks of evaluation of usability that are often laborious and repetitive. The basic principle of this approach promotes the use of recording and analysis of log data in a model-based environment. The results described in this paper show that the log data at model level can be used not only to identify usability problems but also to identify changes to these models in order to correct the encountered problems. This approach is integrated in a process and is supported by a model-based CASE tool for modeling, simulating and evaluating interactive systems. The case study illustrates the principles of the approach and operation of the tool on an interaction technique. It shows how the analysis of log data allows the designer to easily tune the interaction technique (as the results of the analysis of log data are presented at the same abstraction level than models). It proposes an alternative to user tests that are very difficult to configure and to interpret especially when advanced interaction techniques are concerned.

KEYWORDS: Interactive systems modeling, interaction techniques, performance evaluation, model-based usability evaluation.

INTRODUCTION

L'utilisation de modèles pour la conception de systèmes informatiques fournit une description du système abstraite par rapport à son implémentation. De nos jours, ce type d'approche est très largement développé dans le domaine de l'ingénierie logicielle via l'ingénierie dirigée par les modèles [22] qui a émergé des standards UML [25]. En effet, comme ils fournissent une description plus abstraite du système que le code d'implémentation, ils fournissent aussi une opportunité unique pour les différentes parties prenantes (concepteurs, utilisateurs, développeurs…) pour commenter et proposer des modifications aux systèmes pendant la conception.

Dans la communauté IHM, de nombreux chercheurs ont décrit les éléments de l'interface utilisateur à l'aide de modèles. Le lecteur intéressé pourra trouver un état de l'art structuré des modélisations d'interface utilisateur dans [23] où les différentes techniques de modélisation ont été catégorisées selon des critères tels que : le *Langage* (basé sur des réseaux de Petri, basé état, basé flux, basé code et basé sur des contraintes.), la *couverture de l'interaction* (Bas niveau, Multimodalité, Tangible, Fusion, Widget, Rendu et Dialogue), *le passage à l'échelle*

(Exemple jouet, Etude de cas, Application de taille réelle), *le support d'un outil, l'expressivité* (Description des données, Représentation des états, des événements, représentation du *Temps* (Quantitatif ou Qualitatif), *Les comportements concurrents* et *l'Instanciation dynamique)*.

Modèles et utilisabilité

Au-delà de cet aspect modélisation, les modèles peuvent aussi être utilisés pour supporter la vérification de propriétés (comme la vivacité, la sûreté ou plus rarement l'utilisabilité). L'évaluation d'utilisabilité est une préoccupation majeure en IHM car elle est l'un des seuls moyens pour évaluer/ assurer la correspondance entre un système en cours de conception et les besoins et les caractéristiques des utilisateurs. Il est donc logique de trouver des travaux de recherche à l'intersection de ces deux domaines comme l'évaluation d'utilisabilité à distance basée sur les modèles telle que: MultiDevice RemUsine [29] pour l'évaluation distante d'interfaces mobiles, EvaHelper [4] un Framework pour aider l'utilisateur à effectuer une évaluation sur les applications mobiles, ou ReModEl [8] composé d'un serveur, un client (pour le testeur) et une autre interface client pour l'expert en utilisabilité.

En outre des travaux de recherche suivant la même philosophie peuvent aussi être trouvés dans le domaine du web comme l'évaluation automatique à distance d'applications web basé sur une combinaison de monitoring de browser web et des modèles de tâches de l'application (comme dans [28]) ou dans AWUSA [30]. L'évaluation de l'utilisabilité peut aussi être trouvée à des fins plus génériques (autres que mobile ou web) comme dans [10] couplant MeMo pour l'évaluation automatisée de l'utilisabilité et un Framework MASP orienté modèle pour la génération d'interface utilisateur.

Ceci est similaire à [17] qui propose une évaluation de l'utilisabilité automatique (basée sur une spécification d'interface utilisateur structurée) ou à [0] qui présente des méthodes conformes aux architectures dirigées par les modèles (MDA) pour améliorer l'utilisabilité des logiciels par transformation de modèles. Enfin, [14] propose l'intégration de l'analyse des informations d'utilisabilité à partir des événements de l'interface utilisateur à l'aide d'un modèle de composition.

Contributions IHM aux techniques d'interaction

Depuis les travaux de Fitts [11], de nombreuses extensions et raffinements à cette loi ont été proposées comme l'extension de la loi de Fitts pour les tâches bidimensionnelles [21] ou la steering law pour les mouvements contraints (par exemple tâches de trajectoire [2]). Ces lois permettent de prédire la difficulté de réalisation d'une tâche interactive sur un système et peuvent être utilisées à la fois dans la phase de conception ainsi que celle d'évaluation. Lors de la conception, elles peuvent aider à déterminer la représentation graphique des objets (comme la taille et la forme par exemple) ainsi que la technique d'interaction pour manipuler ces objets (comme le multiclic, le drag and drop, …). Ces lois sont à l'origine de nouvelles techniques d'interaction comme le Drag&Pop et le Drag&Pick [5] qui étendent la technique d'interaction drag and drop standard en rapprochant les icônes réactives dès qu'un événement drag est produit. Boomerang [16] propose une autre extension au drag and drop en définissant un mécanisme permettant de suspendre et de reprendre un drag and drop. Une autre extension aux techniques d'interaction standard est le pointage sémantique [7] qui fait varier la vitesse de déplacement de la souris en fonction des objets survolés. D'autres extensions ont aussi été proposées prenant en compte des systèmes d'entrée et de sortie non standard comme le MAGIC pointing [31], ou, pour les écrans de très grande taille, le Drag&Pop [9], le Drag&Throw et le Push&throw [13].

Lorsqu'ils proposent ces techniques d'interaction améliorées, les chercheurs construisent généralement un prototype l'implémentant, et effectuent des tests d'utilisabilité afin d'évaluer la performance de ces nouvelles techniques par rapport aux techniques d'interaction plus standard. Ce genre d'activité soulève six problématiques que nous allons aborder dans la suite de l'article :

Problématique 1 : La Définition du comportement précis de la technique d'interaction : habituellement la technique d'interaction est seulement décrite de façon informelle en utilisant du texte, des captures d'écran ou vidéo (comme la page du drag and pop [32] ou boomerang dans la digital library d'ACM [33]). Ce type de description informelle laisse de nombreuses informations sous-spécifiées comme par exemple dans le cas du drag and pop, la taille des icônes, la distance entre l'icône réactive et l'icône sélectionnée… Avec des techniques d'interaction plus conventionnelles (comme le drag and drop), les mêmes questions restent, laissant des informations importantes non précisément définies.

Problématique 2 : S'assurer de l'exactitude de la technique d'interaction : tout comme pour les tests logiciels, les tests de techniques d'interaction sont généralement effectués de manière informelle par le biais d'une séquence de tests. Comme le comportement de la technique d'interaction dépend de l'état de l'interaction et du comportement de l'utilisateur, la couverture des tests reste très limitée (eu égard à la grande quantité de cas possibles), ne fournissant aucun moyen de s'assurer que la technique d'interaction fonctionnera de façon fiable.

Problématique 3 : L'ajustement de la technique d'interaction pour obtenir des performances optimales : un exemple clair de cet ajustement est l'accélération du pointeur de la souris en fonction de la vitesse de déplacement de la souris sur la table. Une accélération efficace améliore considérablement le temps de sélection, mais elle reste généralement cachée [15]. Chaque fois qu'une nouvelle technique d'interaction est définie, l'identification des éléments ajustables et comment ils doivent être ajustés sont essentiels pour évaluer son utilisabilité. Sans une description précise de la technique d'interaction, ce réglage est effectué au niveau du code et est en général absent de la description de la technique d'interaction rendant très difficile la réutilisation des résultats.

Problématique 4 : L'enregistrement des informations pendant les tests d'utilisabilité : lorsque l'on teste les techniques d'interaction, il est très difficile d'enregistrer des informations pertinentes car les informations recueillies par les outils sont généralement à un faible niveau d'abstraction (événements). Ceci est encore plus critique lorsque l'on essaye de tester des techniques d'interaction multimodales impliquant plusieurs périphériques d'entrée comme un clic combiné lors d'une interaction bimanuelle pour laquelle une vision intégrée des événements est nécessaire.

Problématique 5 : Interprétation des tests d'utilisabilité : les résultats des tests d'utilisabilité d'une technique d'interaction sont difficiles à interpréter car ces résultats se basent en général sur les événements produits. Afin d'améliorer (ajuster ou bien modifier) la technique d'interaction il est nécessaire de pouvoir faire (rapidement à cause des nombreuses boucles d'itération) le lien entre ces événements et les différents états du comportement de la technique d'interaction. Une erreur dans l'établissement de ce lien (entre résultats d'évaluation et comportement de la technique d'interaction) peut avoir des coûts élevés nécessitant par exemple plus de tests d'utilisabilité que ce qui aurait été nécessaire pour définir et mettre au point la technique d'interaction.

Problématique 6 : Réutilisation et raffinement de techniques d'interaction: fournir une description précise (en plus du prototype informatique lui-même) et, éventuellement des vidéos, rend possible à d'autres chercheurs et / ou aux entreprises de réutiliser ce qui a été proposé voire même de l'améliorer. Cela favorise ainsi la réutilisation de travaux antérieurs pour la construction de nouveaux résultats ainsi que la diffusion de connaissances à l'intérieur du domaine de recherche et vers d'autres domaines.

La section suivante présente une approche basée sur des modèles mettant l'accent sur l'évolutivité et la modifiabilité de modèles pour être en mesure de prendre en compte les préoccupations d'utilisabilité. Cette approche apporte des solutions aux problématiques énumérées ci-dessus qui seront exemplifiées sur une étude de cas dans les sections suivantes.

L'approche proposée utilise la notation ICO pour décrire le comportement de l'ensemble des techniques d'interaction. Au-delà des bénéfices fournis par un tel modèle explicite de haut niveau, nous utilisons l'outil PetShop pour exécuter les modèles, et donc d'utiliser le modèle pour intégrer la technique d'interaction modélisée dans un système interactif. Par rapport aux travaux précédents, nous avons intégré dans PetShop un outil de log qui permet, lors de l'exécution des tests d'utilisation, d'enregistrer l'ensemble des évolutions des modèles à la fois au niveau des événements et des états. Ce log est ensuite exploité pour évaluer la performance de la technique d'interaction et identifier directement si des changements doivent être effectués dans le modèle afin d'améliorer la technique d'interaction. Le log permet aussi de déterminer plus facilement où (dans le modèle) ces changements éventuels doivent être réalisés.

Cette approche et le processus associé (prochaine section) sont illustrés directement sur une étude de cas qui présente successivement les principes de modélisation, les principes de l'analyse formelle, les aspects de simulation et d'exploitation du log. Cet article se termine par une discussion sur les avantages et les limites de l'approche.

L'APPROCHE ET ETUDE DE CAS

Cette section présente un processus (voir Figure 1) pour la conception, la simulation, le log, l'analyse et l'évaluation de technique d'interaction. Ce processus est directement présenté sur une étude de cas qui permet de rendre explicite pour chaque phase les informations en entrée, l'activité à réaliser dans la phase et les informations produites qui seront utilisées dans les phases suivantes.

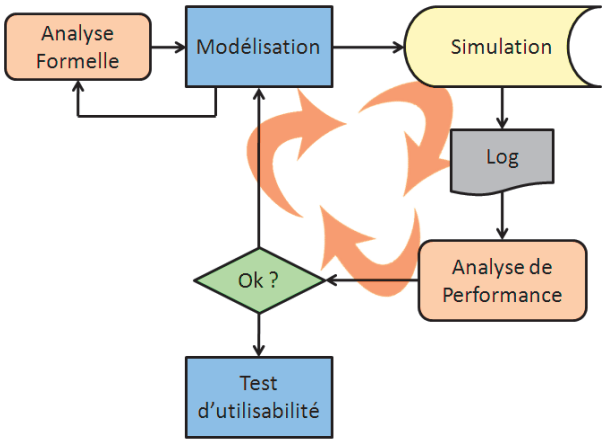

Figure 1. Processus pour l'ingénierie logicielle de techniques d'interaction.

Ce processus commence par **l'activité de modélisation** durant laquelle le modèle de la technique d'interaction est construit en utilisant le formalisme ICO [23]. Ce modèle décrit de façon complète et non ambigüe les états de cette technique d'interaction ainsi que les événements déclenchant les changements d'états.

La **phase de simulation** est rendue possible par l'outil PetShop qui permet l'exécution de modèles ICOs. Plus d'information sur l'outil est disponible ici [3].

Lors de la simulation, les évolutions sur le modèle sont enregistrées automatiquement dans un fichier de log (**phase de log**) qui est exploité dans la phase suivante pour **évaluer la performance** de la technique d'interaction. Si la performance est compatible avec ce qui est attendu, la technique d'interaction peut être testée avec des utilisateurs (**phase de Test d'utilisabilité**). Si ce n'est pas le cas, les informations du log sont utilisées pour modifier le modèle ICO décrivant la technique d'interaction (flèche "not ok" dans la Figure 1). Ce processus de conception dédié à la conception à base de modèle s'intègre parfaitement dans les processus de conception en Interaction Homme-Machine standards si la technique d'interaction est considérée comme une partie intégrante du système interactif. Par contre ce processus est directement intégrable dans des processus comme ce-

lui de [12] qui représente explicitement 2 phases une pour le système interactif et une pour la technique d'interaction.

Modélisation et étude de cas

Dans cette application (Figure 2) un ensemble d'icônes est présenté dans une fenêtre sur une grille. Ces icônes peuvent être déplacés à différents endroits. Le but de l'utilisateur est d'enlever toutes les icônes de l'interface à l'aide de deux techniques d'interaction.

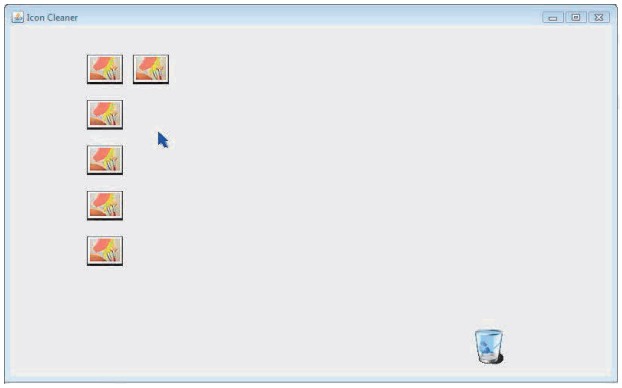

Figure 2. Interface utilisateur de l'étude de cas

La première appelée Pointage Sémantique correspond à la technique d'interaction présentée dans [7]. La seconde est une technique d'interaction de type Drag&Pop [5]. Ces techniques ont pour but d'améliorer le Drag and Drop simple en accroissant les performances des utilisateurs dans l'accomplissement de cette tâche. On remarquera que ces différentes techniques d'interaction présentent certains comportements semblables. En effet, chacune nécessite la sélection d'une icône, son déplacement et enfin son dépôt sur un objet réactif (comme la poubelle). Les modèles de ces deux techniques d'interaction ont été réalisés suivant le processus décrit ci-dessus et modélisées avec la notation ICO. Toutefois, pour des raisons de place nous ne présentons ici que le modèle de la technique d'interaction Pointage Sémantique qui est présentés sur les Figure 3. Cette similarité entre techniques d'interaction est très facilement visible sur les modèles et correspondent aux comportements similaires.

Le modèle de la Figure 3 contient toutes les informations à propos de l'état courant de la technique d'interaction ainsi que l'ensemble des événements disponibles pour l'utilisateur, leurs séquences autorisées et leurs impacts sur l'état.

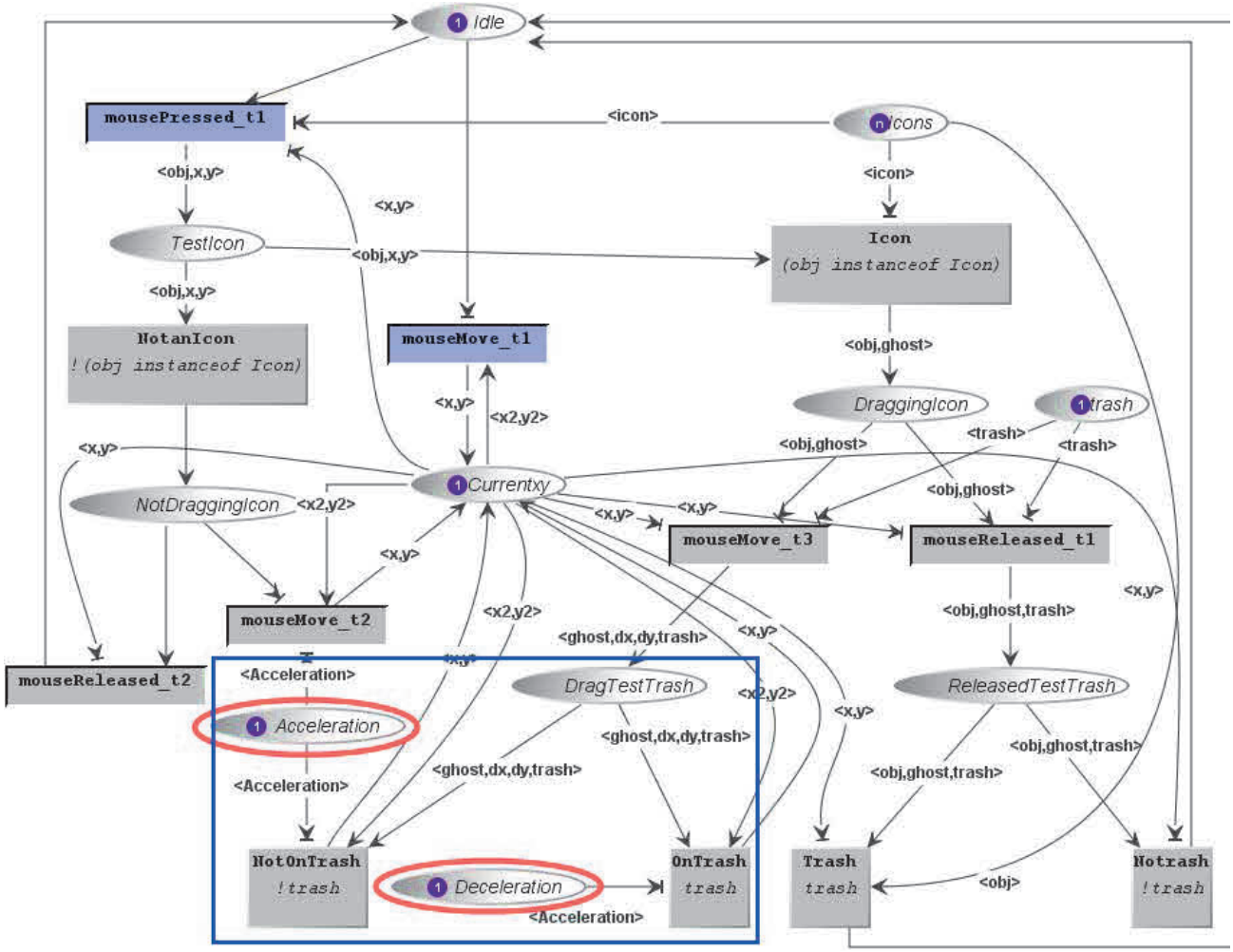

Figure 3. Modèle ICO de la technique d'interaction du Drag&Drop avec pointage sémantique

A l'état initial, la technique d'interaction est dans l'état d'attente (il y a un jeton dans la place *Idle*) et la position x, y du curseur est contenue dans un jeton dans la place *Currentxy*, les références aux objets graphiques sont stockées dans la valeur des jetons de la place *Icons*. Enfin la référence à l'objet poubelle est stockée dans la va-

leur du jeton contenu dans la place *Trash*. A partir de cet état initial, deux transitions sont disponibles (représentées par le cadre dans le modèle: mouseMove_t1 and mousePressed_t1. La transition mouseMove_t1 est franchie lorsque l'événement mouseMove correspondant est produit lors d'une action de l'utilisateur sur la souris. A ce moment, la valeur stockée dans le jeton de la place *Currentxy* est mise à jour et contient la nouvelle position du curseur.La transition mousePressed_t1 est franchie quand l'utilisateur presse le bouton de la souris.

A ce moment, le modèle teste si le curseur est sur une icône ou non (ce test est représenté par les transitions NotanIcon and Icon). Si la transition NotanIcon est franchie alors le modèle attend que le bouton de la souris soit relâché (lors de la réception de cet événement, la transition mouseReleased_t1 est franchie et le modèle revient à son état initial en déposant un jeton dans la place *Idle*). Par contre, si le curseur est sur une icône, le modèle attend les événements de déplacement de la souris (la transition mouseMove_t3) ou un événement mouseReleased. Lors de la réception d'un événement mouseMove_t3, un jeton est déposé dans la place *DragTestTrash* et deux transitions permettent de vérifier si le pointeur est sur un objet réactif (ici la poubelle) ou non. Grâce à cette information, il est possible de changer la vitesse du pointeur lorsque celui-ci est sur l'objet réactif pour ralentir son déplacement. Ceci rend plus difficile de déplacer le curseur de la souris au-delà de la cible et augmente la performance des utilisateurs comme expliqué dans [7]. La place *Acceleration* (resp. *Deceleration*) (entourées en Figure 3) est liée aux transitions MouseMove_t2 et NotOnTrash (resp. à la transition OnTrash). La place *Acceleration* contient une valeur utilisée pour l'accélération du curseur lorsqu'un objet est déplacé (et qu'il n'est pas sur une icône) rendant le déplacement du "fantôme" de l'icône plus rapide que le déplacement du périphérique d'entrée sur la table. Le fait de pouvoir changer cet effet en modifiant uniquement la valeur du jeton de la place *Acceleration* correspond à la notion d'ajustement de la technique d'interaction qui correspond à la problématique 3 de l'introduction.

Lorsque l'événement mouseReleased est reçu, le modèle teste si la position du curseur est sur l'icône de la poubelle (transition Trash) ou non (transition Notrash). Si oui, le fichier est effacé (ce qui est modélisé dans le code de la transition Trash non représenté sur la figure). Que la transition Trash ou la transition NoTrash soit franchie, un jeton est déposé dans la place *Idle* et le modèle revient à son état initial. On peut remarquer que le modèle définit entièrement la technique d'interaction à la fois en termes d'états atteignables, d'événements autorisés et de comportement graphique. Ceci correspond aux problèmes soulevés par la problématique 1 de l'introduction.

Analyse Formelle

Les fondements réseaux de Petri du formalisme ICO permettent d'utiliser les techniques d'analyse de propriétés des réseaux de Petri tels que les invariants de place et de transition, les trappes et siphons [19] ou les techniques de réduction [6] pour vérifier des propriétés sur les modèles.

L'analyse d'invariants permet de prouver des propriétés telles que la vivacité de transition. Dans le cas de techniques d'interaction, cela peut permettre de prouver par exemple que la transition gérant le déplacement de la souris sera toujours disponible (donc que l'événement ne sera jamais ignoré) ce qui est compatible avec le fait qu'il est toujours possible de produire ce type d'événement en déplaçant la souris. Selon les résultats obtenus par l'analyse, il peut être décidé de modifier le modèle. Le lecteur intéressé pourra trouver une explication détaillée sur la façon de réaliser ce type de vérification pour les systèmes interactifs dans [26].

Simulation des modèles

Grâce à l'environnement PetShop, les modèles peuvent être exécutés et simulés. Cela signifie que les événements produits par l'utilisateur lorsqu'il interagit avec les périphériques d'entrée sont directement reçus par les modèles en cours d'exécution qui évolueront en conséquence (en fonction de chaque événement reçu et de l'état des modèles au moment où l'événement est reçu). Ceci est vrai pour l'ensemble des modèles qu'ils représentent le système interactif ou les techniques d'interaction (voir [3] pour plus de détail sur cet aspect). Dans l'environnement PetShop un outil spécifique a été développé pour tracer toutes les évolutions des modèles à l'exécution. Cet outil (appelé outil de log) enregistre les diverses évolutions du modèle (changement de valeur des jetons, franchissement de transitions, déplacements de jeton, …) dans un fichier de log (voir détail dans la section suivante). Le fait que ces modèles soient exécutables et dirigent l'exécution de l'application est une condition nécessaire pour être capable de réaliser un enregistrement des données log basées sur les modèles et permettre leur analyse qui est présentée dans la suite de cet article. Cet outil de log est particulièrement utile pour tester le système interactif. En complément des techniques de vérification formelle présentées dans la section précédente, ces outils apportent des solutions au problème de robustesse des techniques d'interaction identifié dans la problématique 2.

Enregistrement des évolutions des modèles (Log)

Durant l'exécution des modèles, un fichier de log (présenté en Figure 4) est produit contenant les informations sur l'évolution des modèles. Les informations enregistrées sont : le franchissement d'une transition, l'ajout ou le retrait d'un jeton d'une place, et le changement de substitutions (non présenté ici). Ces données ont vocation à être utilisées pour évaluer la performance d'une technique d'interaction. Si la performance ne correspond pas aux attentes, les données du log peuvent servir à modifier ou ajuster le modèle qui sera ensuite à nouveau simulé. Ces modifications ou ajustements sont facilités par le fait que le log est basé sur la structure du modèle. Ceci est fortement différent du log classique comme dans [14] où les informations enregistrées sont uniquement relatives aux actions utilisateurs sur l'interface (clic, déplacement) ou sur les périphériques d'entrée. Cet enregistrement de l'évolution de nos modèles et le fait que les modifications éventuelles portent aussi sur les modèles

(donc se situent au même niveau d'abstraction) sont des éléments de réponse à la problématique 4 de l'introduction.

La Figure 4 présente un extrait du log produit lors de l'utilisation de l'application de l'étude de cas avec la technique d'interaction décrite sur la Figure 3. Il correspond à la sélection d'une icône, le déplacement de l'icône vers la poubelle et relâchement de l'icône sur la poubelle. Chacune des actions commence par le franchissement d'une transition correspondant aux actions utilisateurs (type=*transition*, action=*fire)* suivi d'un déplacement de jeton (type=*place*, action=*token_added /token_removed*) dû au franchissement de cette transition. Les trois première lignes du tableau représentent un *mouseMove_t1* où un jeton est enlevé de la place *currentxy* avec comme valeur <124,239> (dans la colonne token de la ligne 02) et un nouveau jeton est déposé dans cette même place avec la valeur <128,240> (ligne 03). Cette transformation de valeur se retrouve dans la colonne substitution de la ligne de franchissement de transition *mouseMove_t1* (ligne 01) ce qui signifie que cette valeur du jeton est utilisée par la transition. Ensuite lors du franchissement de *mousePressed_t1* (ligne 04), le jeton de la place *Idle* est enlevé (ligne 05) et un jeton est ajouté dans la place *testIcon* (ligne 06) avec comme valeur la référence à l'objet sélectionné (icon.Icon[.. dans la colonne token) obtenue lors du tir de la transition (colonne substitutions de *mousePressed_t1* ligne 04). On peut voir dans la colonne Time présentant le temps en seconde arrondi à 3 décimales que ce mousePressed_t1 à eu lieu à 1,353 seconde du début du test.

	Type	Name	Action	Time	Multiplicity	Token	Subsitutions
01	transition	mouseMove_t1	fire	1,279			1*{y2=> 239, x2 => 124, y => 240, x => 128}
02	place	currentxy	token_removed	1,279	1	<124,239>	
03	place	currentxy	token_added	1,279	1	<128,240>	
04	transition	mousePressed_t1	fire	1,353			1*{obj => icon.Icon[,100,210,48x48,align
05	place	Idle	token_removed	1,353	1	<>	
06	place	testIcon	token_added	1,353	1	<icon.Icon[,100,210,48x48...	
07	transition	Icon	fire	1,358			1*{offsetx => 28, offsety => 30, obj =>
08	place	testIcon	token_removed	1,358	1	<icon.Icon[,100,210,48x48...	
09	place	DragIcon	token_added	1,358	1	<icon.Icon[,100,210,48x48...	
10	transition	mouseMove_t3	fire	1,428			1*{offsetx => 28, offsety => 30, dx => 2, dy => 1, obj =>
11	place	DragTestTrash	token_added	1,428	1	<icon.Ghost[,100,210,48x48	
12	transition	NotOnTrash	fire	1,436			1*{offsetx => 28, offsety => 30, dx => 2, dy => 1, y2 =>
13	place	DragTestTrash	token_removed	1,436	1	<icon.Ghost[,100,210,48x48	
14	place	currentxy	token_removed	1,436	1	<128,240>	
15	place	currentxy	token_added	1,436	1	<134,243>	
16	transition	mouseMove_t3	fire	2,103			1*{offsetx => 28, offsety => 30, dx => -3, dy => -2, obj =>
17	place	DragTestTrash	token_added	2,103	1	<icon.Ghost[,594,423,48x48	
18	transition	NotOnTrash	fire	2,11			1*{offsetx => 28, offsety => 30, dx => -3, dy => -2, y2 =>
19	place	DragTestTrash	token_removed	2,11	1	<icon.Ghost[,594,423,48x48	
20	place	currentxy	token_removed	2,11	1	<622,453>	
21	place	currentxy	token_added	2,11	1	<613,447>	
22	transition	mouseMove_t3	fire	2,178			1*{offsetx => 28, offsety => 30, dx => -4, dy => -6, obj =>
23	place	DragTestTrash	token_added	2,178	1	<icon.Ghost[,585,417,48x48	
24	transition	OnTrash	fire	2,187			1*{offsetx => 28, offsety => 30, dx => -4, dy => -6, y2 =>
25	place	DragTestTrash	token_removed	2,187	1	<icon.Ghost[,585,417,48x48	
26	place	currentxy	token_removed	2,187	1	<613,447>	
27	place	currentxy	token_added	2,187	1	<609,441>	
28	transition	mouseMove_t3	fire	2,403			1*{offsetx => 28, offsety => 30, dx => 5, dy => -1, obj =>
29	place	DragTestTrash	token_added	2,403	1	<icon.Icon[,100,210,48x48...	
30	transition	OnTrash	fire	2,412			1*{offsetx => 28, offsety => 30, dx => 5, dy => -1, y2 =>
31	place	DragTestTrash	token_removed	2,412	1	<icon.Icon[,100,210,48x48...	
32	place	currentxy	token_removed	2,412	1	<614,434>	
33	place	currentxy	token_added	2,412	1	<619,433>	
34	transition	mouseReleased_t1	fire	2,702			1*{offsetx => 28, offsety => 30, obj =>
35	place	DragIcon	token_removed	2,702	1	<icon.Icon[,100,210,48x48...	
36	place	ReleasedTestTrash	token_added	2,702	1	<icon.Icon[,100,210,48x48...	
37	transition	Trash	fire	2,707			1*{offsetx => 28, offsety => 30, trash =>
38	place	Icons	token_removed	2,707	1	<icon.Icon[,100,210,48x48...	
39	place	ReleasedTestTrash	token_removed	2,707	1	<icon.Icon[,100,210,48x48...	
40	place	Idle	token_added	2,707	1	<>	

Figure 4. Extrait du log basé modèle

On vérifie ensuite si l'objet est une icône (franchissement de la transition *Icon* ligne 07). Lors d'un déplacement (*mouseMove_t3* en ligne 10), le modèle teste si le curseur est sur la poubelle ou non. Comme il n'est pas encore sur la poubelle, il y a franchissement de *NotOntrash* (ligne 12) et un enlèvement puis ajout de jeton dans la place *currentxy* (lignes 14&15) pour mettre à jour la valeur du jeton dans cette place (passage de la valeur <128,240> à la valeur <134,243>). Une partie du log est ensuite ignorée dans cette présentation. Elle représente une suite de déplacements et de franchissements de *NotOnTrash* comme les cinq lignes précédentes (lignes 11 à 15) que l'on retrouve dans les lignes suivantes (lignes16 à 21). Lorsqu'un déplacement (*mouseMove_t3* ligne 22) positionne le curseur sur la poubelle, la transition *OnTrash* est franchie (ligne 24) et de la même façon que lors du

franchissement de *NotOntrash*, la place *currentxy* est mise à jour (lignes 26&27). Une suite de déplacements avec le curseur sur la poubelle correspond aux lignes 22 à 27. On retrouve le dernier déplacement de ce type aux lignes 28 à 33. Un événement de l'utilisateur relâchant le bouton de la souris est ensuite reçu et la transition *mouseReleased_t1* est franchie (ligne 34) retirant le jeton de la place *DragIcon* (ligne 35) et le déposant dans la place *ReleasedTestTrash* (ligne 36). Etant donné que le curseur est sur la poubelle, la transition *Trash* est franchie (ligne 37) à 2,707secondes (colonne Time) du début du test et le jeton de la place *ReleasedTestTrash* (ligne 39) est retiré. Le jeton correspondant à l'icône est retiré de la liste des icônes dans la place *Icons* (ligne 38) et un jeton est déposé dans la dans place *Idle* (ligne40) replaçant le modèle dans son état initial.

Nous pouvons calculer le temps de déplacement dans le cas du pointage en soustrayant les temps lors du franchissement de la transition *mousePressed_t1* (ligne 04) au temps de franchissement de la *transition* Trash (ligne 37) ce qui nous donne 1,354 secondes. Nous pouvons également calculer le nombre d'erreurs qui sont intervenues lors de cette interaction. Le nombre d'erreur est obtenu en comptabilisant le nombre de fois ou l'utilisateur manque la cible (dans ce cas la poubelle). Dans notre modèle ceci correspond au nombre de fois où la transition notOntrash est franchie après une transition OnTrash. En effet, cela représente le fait que l'utilisateur a dépassé la poubelle et doit donc agir en conséquence et revenir à un état antérieur (le curseur sur la poubelle) avant de relâcher le bouton de la souris. Cette interaction est exécutée plusieurs fois avec des valeurs d'accélération différentes et les données sont récupérées grâce à l'outil de log durant la simulation dans PetShop.

L'analyse des données obtenues par le log nous permet de calculer l'évolution du nombre d'erreur et le temps d'exécution de la tâche. Ces résultats sont rassemblés et présenté dans la Figure 5. Ce graphique montre que lorsque l'accélération du pointeur (valeur du jeton dans la place *Acceleration* de la Figure 3) augmente, la performance est meilleure. Mais à partir d'un taux d'accélération de 5, les erreurs deviennent trop importantes et le temps pour réaliser le Drag and Drop augmente et devient plus grand que pour la technique d'interaction sans aucun accélération (début de la courbe).

Les barres du diagramme (axe de droite) représentent le nombre moyen d'erreur sur 10 essais pour un Drag and drop, la courbe représente le temps d'exécution d'un drag and drop (axe de gauche de 0,6s à 1,4s). L'abscisse représente le taux d'accélération. Cette analyse des résultats permet de répondre à la problématique 5 en reliant aisément les résultats de l'évaluation avec les éléments du modèle sur lesquels doivent porter les modifications.

Pour des raisons de place, nous n'avons pas présenté sur ce diagramme l'impact de la décélération lorsque le curseur souris est sur l'icône cible. Ceci contrebalance l'impact de l'accélération et la performance est significativement améliorée.

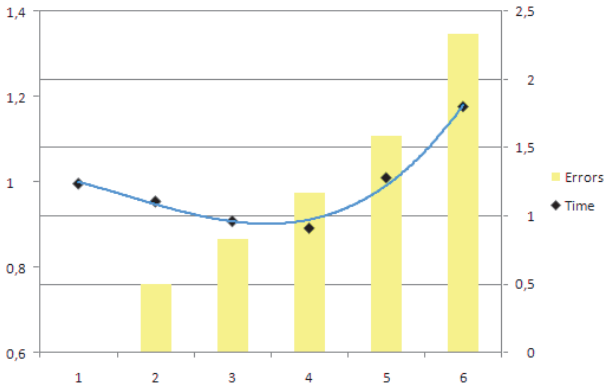

Figure 5. Résultat de l'analyse de log du pointage sémantique combiné à l'accélération du curseur

DISCUSSION ET CONCLUSION

Dans cet article nous avons présenté une approche pour permettre l'ingénierie de techniques d'interaction à base de modèles. Cette approche permet de modéliser, tester et évaluer différentes techniques d'interaction qui vont bien au-delà (en terme de complexité) des techniques d'interaction standard. Nous avons montré comment cette approche peut apporter des solutions à un ensemble de problèmes qui se posent lorsqu'on s'intéresse à l'ingénierie des techniques d'interaction. En particulier nous avons montré comment l'évaluation basée sur modèles (au moyen de logs) rend possible d'ajuster finement différents paramètres de la technique d'interaction.

Cette approche a été illustrée à l'aide d'une étude de cas où nous avons montré l'intérêt d'une évaluation de l'utilisabilité basée sur les modèles. Ces évaluations montrent des résultats bien connus dans le domaine de l'IHM comme le fait que l'accélération augmente l'efficacité pour la sélection d'une cible jusqu'à un certain point. L'objectif de cette approche est de l'appliquer sur des techniques d'interaction plus récentes et plus sophistiquées (incluant les multimodales) qui sont plus difficile à valider, spécialement par les tests utilisateurs comme celles déjà modélisées dans [8].

Le domaine ciblé est celui des cockpits interactifs où les standards tels que la spécification ARINC 661 fournissent seulement des recommandations pour les interfaces WIMP en laissant un champ entier d'investigation lorsque les techniques d'interaction de manipulation directe sont concernées. Ce domaine cible fournit des contraintes fortes en termes de modélisation et d'évaluation de techniques d'interaction dans la mesure où les performances utilisateur peuvent avoir des conséquences catastrophiques pour les opérations et dans la mesure où l'ensemble des éléments logiciels sont à spécifier et développer.

REMERCIEMENTS

Le travail présenté dans cet article est en partie financé par l'action R&T CNES Tortuga R-S08/BS-0003-029 et le contrat Airbus CIFRE PBO D08028747–788/2008.

REFERENCES

1. Abrahão S., Iborra E., and Vanderdonckt J. 2007. Usability Evaluation of User Interfaces Generated with a Model-Driven Architecture Tool. In Maturing Usability: Quality in Software, Interaction and Value, Series: Springer Human-Computer Interaction Series, Vol. 10, 610 p., 2007, Springer.
2. Accot, J. and Zhai, S. 1997. Beyond Fitts' law: models for trajectory-based HCI tasks. In Proceedings of CHI'97 (Atlanta, USA). ACM, pp. 295-302.
3. Bastide, R., Navarre, D., and Palanque, P. 2002. A model-based tool for interactive prototyping of highly interactive applications. In Proceedings of CHI '02, pp. 516-517.
4. Balagtas-Fernandez F., Hußmann H., A Methodology and Framework to Simplify Usability Analysis of Mobile Applications In Proceedings of ASE 2009, pp. 520-524.
5. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P. Bederson, B., and Zierlinger, A. Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In Proceedings of Interact 2003 (Zurich Switzerland), pp. 57-64.
6. Berthelot G. Transformations and Decompositions of Nets; LNCS 254, pp. 359-376, Springer Verlag 1987.
7. Blanch R., Guiard Y. and Beaudouin-Lafon M. Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation. In Proceedings of CHI 2004, (Vienna, Austria, April 2004), pp. 519-526.
8. Buchholz G, Engel J, Märtin C, Propp S. Model-based usability evaluation - evaluation of tool support. HCII 2007, (Beijing, China) pp. 1043-52. Springer-Verlag.
9. Collomb, M. and Hascoët, M. 2008. Extending drag-and-drop to new interactive environments: A multi-display, multi-instrument and multi-user approach. Interact. Comput. 20, 6 (Dec. 2008), pp. 562-573.
10. Feuerstack, S., Blumendorf, M., Kern, M., Kruppa, M., Quade, M., Runge, M., and Albayrak, S. 2008. Automated Usability Evaluation during Model-Based Interactive System Development. In Proceedings of TAMODIA'08 (Pisa, Italy). LNCS, vol. 5247. Springer-Verlag, pp. 134-141.
11. Fitts, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, 47, pp. 381-391.
12. Gulliksen J. & Goransson B. Usability Design: Integrating User Centered System Design in Software Development Process. In proceeding of IFIP TC 13 INTERACT 2003 conference, Zurich, Switzerland.
13. Hascoët, M. 2003. Throwing models for large displays. In: HCI2003, Designing for society, vol. 2. British HCI Group. pp. 73-77.
14. Hilbert, D. M. and Redmiles, D. F. 2000. Extracting usability information from user interface events. ACM Comput. Surv. 32, 4 (Dec. 2000), pp. 384-421.
15. Hinckley, K., Cutrell, E., Bathiche, S., and Muss, T. 2002. Quantitative analysis of scrolling techniques. In Proceedings of CHI'02 (Minneapolis, USA), pp. 65-72.
16. Kobayashi, M. and Igarashi, T. 2007. Boomerang: suspendable drag-and-drop interactions based on a throw-and-catch metaphor. In Proceedings of UIST'07, ACM, 187-190.
17. Kristoffersen, S. 2009. A Preliminary Experiment of Checking Usability Principles with Formal Methods. In Proceedings of ACHI'09. IEEE Computer Society, 261-270
18. Ladry, J., Navarre, D., and Palanque, p. 2009. Formal description techniques to support the design, construction and evaluation of fusion engines for sure (safe, usable, reliable and evolvable) multimodal interfaces. In Proceedings of the 2009 international Conference on Multimodal interfaces (Cambridge, Massachusetts, USA, November 02 - 04, 2009). ICMI-MLMI '09. ACM, New York, NY, 185-192
19. Li Jiao, To-Yat Cheung, and Weiming Lu. Characterizing Liveness of Petri Nets in Terms of Siphon. ICATPN 2002, LNCS 2360, pp. 203-216, Springer-Verlag
20. Lim, K. Y. and Long, J. (1994). The Muse Method for Usability Engineering. Cambridge University Press.
21. MacKenzie, I. S. and Buxton, W. 1992. Extending Fitts' law to two-dimensional tasks. In Proceedings of CHI'92 (Monterey, United States), ACM, pp. 219-226
22. Mellor S. and Balcer M. (2002). Executable UML: A Foundation for Model-Driven Architectures. Addison-Wesley Longman Publishing Co., Inc., Boston, USA, 2002.
23. Navarre, D., Palanque, P., Ladry, J., and Barboni, E. 2009. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM TOCHI 16, 4 (Nov. 2009), pp. 1-56.
24. Navarre D. Palanque P. Bastide R, and Sy, O. Structuring interactive systems specifications for executability and prototypability. In Proceedings of DSV-IS'20 00. LNCS. n° 1946, Springer Verlag, pp. 145-161.
25. Object Management Group (2003). Unified Modelling Language (UML) 2.0 Superstructure Specification, August 2003. Ptc/03-08-02, pp. 455-510
26. Palanque P. & Bastide R. (1995). Verification of an Interactive Software by analysis of its formal specification. Proc. of the IFIP TC 13 Interact'95, pp. 191-197
27. Parnas, D. L. On the use of transition diagrams in the design of a user interface for an interactive computer system. 1969. In Proceedings of 24th National ACM Conf. pp. 379-385.
28. Paternò, F. and Paganelli, L. 2001. Remote automatic evaluation of web sites based on task models and browser monitoring. In Proceedings of CHI '01. ACM, 283-284.
29. Paternò, F., Russino, A., Santoro, C. (2007) Remote evaluation of Mobile Applications. In Proceedings of TAMODIA 2007, (Toulouse, France), LNCS Vol. 4849.
30. Tiedtke, T., Märtin, C., Gerth, N.: AWUSA - A Tool for Automated Website Analysis. In: PreProceedings of the 9th Int. Workshop DSV-IS 2002, Rostock, Germany, pp. 251-266.
31. Zhai, S., Morimoto, C. Ihde, S. (1999) Manual And Gaze Input Cascaded (MAGIC) Pointing. In Proceding of CHI'99, pp. 246-253.
32. Drag and Pop Home page: accessed 10th May 2010 http://patrickbaudisch.com/projects/dragandpop/index.html
33. Boomerang: suspendable drag-and-drop interactions based on a throw-and-catch metaphor. Accessed 10th May 2010 http://portal.acm.org/citation.cfm?id=1294211.1294243