Couplage de Ressources d'Interaction : des Bases pour l'Analyse

Coupling Interaction Resources: Foundations for Analysis

Nicolas BARRALON

Université Joseph Fourier, France
nicolas.barralon@imag.fr

Résumé. Le couplage de ressources d'interaction est l'un des fondements de l'informatique ambiante où l'utilisateur construit son espace par assemblage d'entités. Dans cet article, nous explorons de manière rationnelle cette notion étudiée jusqu'ici de manière exploratoire. Nous proposons comme point de départ un cycle de vie du couplage dont les états devraient êtres observables et contrôlables de manière souple et robuste par l'utilisateur. Nous illustrons notre propos avec l'état de l'art.
Mots-clés. Couplage de ressources d'interaction, technique d'interaction, informatique ambiante.

Abstract. Coupling interaction resources is one of fundament of ubiquitous computing. Users can dynamically build their own interactive space using entities composition. In this article, we will explain rationally this "new" notion which is currently only experimentally explored. As starting point, we propose a coupling life cycle composed by states. These states would be flexibly and robustly observable and manageable. We illustrate our propositions with a state of the art.
Keywords. Coupling interaction resources, interaction techniques, ubiquitous computing.

1 Introduction

Le domaine de l'informatique subit régulièrement de profonds bouleversements provoqués par l'émergence de nouvelles technologies. Aujourd'hui, une nouvelle (r)évolution se prépare avec la convergence des réseaux ad hoc sans fil, des micro et nano systèmes et la maturation de trente années de recherche en systèmes répartis, en perception artificielle et en Interaction Homme-Machine (IHM). Cette convergence de technologies permet d'envisager le passage d'une informatique confinée et statique, à une informatique ambiante et dynamique (Weiser, 1991) favorisant l'émergence d'écosystèmes par couplage de ressources d'interaction.

Une *ressource d'interaction* est une entité physique, support des échanges d'information entre un système informatique et un utilisateur. Le clavier, la souris et l'écran sont les ressources d'interaction types de l'informatique conventionnelle. En informatique ambiante, tout objet familier, comme le lapin (Nabaztag) de la Figure 1, peut être augmenté et servir de ressource d'interaction. De plus, ces ressources

peuvent être assemblées de manière opportuniste. Par exemple, deux amis se rencontrant par hasard assemblent leur PDA en les plaçant côte à côte (Figure 2 à gauche). Ils obtiennent ainsi une surface d'affichage suffisamment grande pour engager une collaboration. Ou encore, un voyageur à proximité d'un mur actif, obtient des renseignements que l'environnement transfère automatiquement au bon format sur le dispositif personnel (Figure 2 à droite). Ces deux exemples montrent que, par couplage de ressources d'interaction, l'utilisateur obtient de nouveaux services.

Figure 1. *Nabaztag, le lapin relié à Internet par Wi-Fi indique par des changements de couleurs, des mouvements des oreilles ou des sons, toutes sortes d'informations. L'informatique ambiante étend cet usage à tous les objets qui nous sont familiers.*

Figure 2. *Exemples de couplage opportuniste de ressources d'interaction.*

Toutefois, le couplage de ressources d'interaction n'est pas un phénomène nouveau : en IHM graphique conventionnelle, la souris et l'écran sont couplés pour fournir le service de pointage. De même, la souris est liée au clavier pour désigner le

focus, lieu d'insertion des caractères. Mais, par conception, ces couplages sont immuables et les classes de ressources d'interaction sont connues par avance. En conséquence, il suffit à l'utilisateur d'apprendre un nombre limité de conventions pour se servir de ces ressources. En informatique ambiante, la diversité des classes de ressources d'interaction et leur composition dynamique ouvrent un large champ d'expérience qui pose, pour le développeur, des problèmes de mise en œuvre technique, et pour l'utilisateur, des difficultés d'appropriation.

Sur le plan technique, des solutions commencent à émerger : la boîte à outils ICon permet de configurer des ressources d'interaction d'entrée selon un modèle à flot de données (Dragicevic, 2004) ; Ubit (Lecolinet, 2003) est capable de gérer plusieurs écrans et plusieurs pointeurs au-dessus de X Window, ou encore I-AM permet la construction dynamique d'un espace d'interaction multi-écran, multi-pointeur, multi-clavier (Lachenal, 2004). Quant à l'utilisateur, comprend-il qu'un couplage est possible ? Peut-il en prédire les effets ? Connaît-il la technique d'interaction pour faire ou défaire un couplage particulier ? Autant de questions que Bellotti et ses co-auteurs posent à propos des systèmes en informatique ambiante (Bellotti *et al.*, 2002).

Dans cet article, nous proposons une analyse rationnelle de la notion de couplage en sorte que la mise en œuvre technique du couplage en informatique ambiante réponde aux critères d'utilisabilité. Dans ce but, nous proposons un cadre d'analyse (section 3) suivi en section 4 de l'introduction d'une notation formelle. Cette représentation du couplage nous permettra de raisonner de manière systématique sur la signification et les effets d'un couplage de ressources d'interaction. Cette analyse théorique sera suivie par une discussion sur les techniques d'interaction permettant à l'utilisateur de faire et défaire les couplages. Nous commençons par présenter quelques exemples de l'état de l'art qui nous serviront à illustrer la discussion.

2 Exemples de l'état de l'art

Nous ferons la distinction entre le couplage de ressources d'interaction conventionnelles comme le clavier et la souris, et des couplages plus innovants.

2.1 Couplages de ressources conventionnelles

Avec Pebbles (Myers *et al.*, 1998), des PDA peuvent être liés dynamiquement à un écran interactif partagé. De façon similaire, Dynamo (Izadi *et al.*, 2003) permet à plusieurs utilisateurs de collaborer via une surface partagée en connectant leur dispositif personnel (ordinateur portable ou PDA) en des « points d'interaction mobiles » (*mobile interaction points*). Avec les ConnecTables du projet I-Land (Tandler *et al.*, 2001), l'utilisateur agrandit son espace d'affichage en mettant en contact le haut des écrans de deux tablettes. Hinckley propose le même type de service par couplage de tablettes (Hinckley, 2003).

La Figure 3 montre l'exemple de deux stations de travail, un PC et un Macintosh, dont les surfaces ont été couplées par les bords droit-gauche. Une fenêtre est affichée à cheval entre les deux écrans. Avec le clavier du PC, l'utilisateur saisit un texte dans l'interacteur affiché sur une surface du Macintosh. Les curseurs des deux machines peuvent migrer sans restriction entre les deux écrans qui forment, grâce à I-AM, un seul espace logique d'affichage. I-AM (Interaction Abstract Machine) se substitue aux gestionnaires de fenêtres conventionnels pour offrir à l'utilisateur, comme au programmeur, un espace d'interaction dynamique multi-écran, multi-clavier, multi-curseur, multi-souris comme si ces ressources étaient gérées par un seul ordinateur.

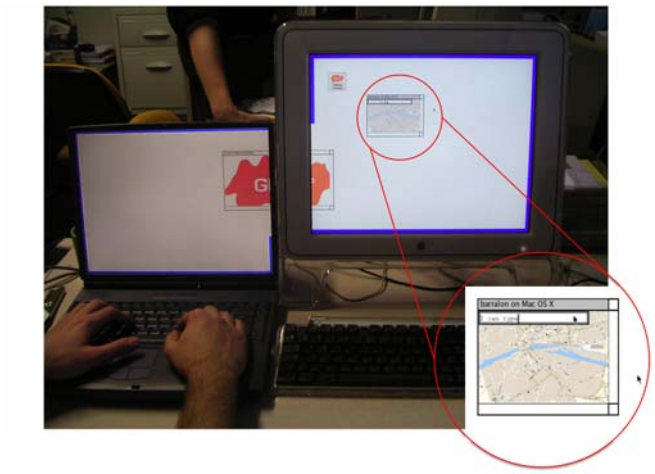

Figure 3. *Un espace d'interaction construit avec I-AM à partir des ressources d'interaction d'un PC portable exécutant Windows XP et d'un MacIntosh exécutant MacOSX.*

2.2 Couplage de ressources non conventionnelles

Comme le montre la Figure 4 les utilisateurs des DataTiles construisent de nouveaux services par assemblage de tuiles sur un écran à cristaux liquides (Rekimoto *et al.*, 2001).

Figure 4. *Les DataTiles (Rekimoto et al., 2001) se présentent sous la forme de tuiles transparentes que l'utilisateur dispose sur une table et sur lesquelles des informations sont rétro-projetées. Chaque tuile est identifiée et joue un rôle particulier (curseur, carte géographique, etc.). L'utilisateur peut les composer, par exemple placer la tuile "météo" sur la tuile "ligne du temps" pour découvrir le temps qu'il fera demain : la topologie des tuiles est un véhicule sémantique.*

Ou encore, avec la Table Magique, l'utilisateur manipule un espace d'information projeté sur une table au moyen de jetons en plastique suivis par un système de vision par ordinateur (Bérard, 2003). Le couplage de deux jetons a lieu par leur mise en contact (voir Figure 5).

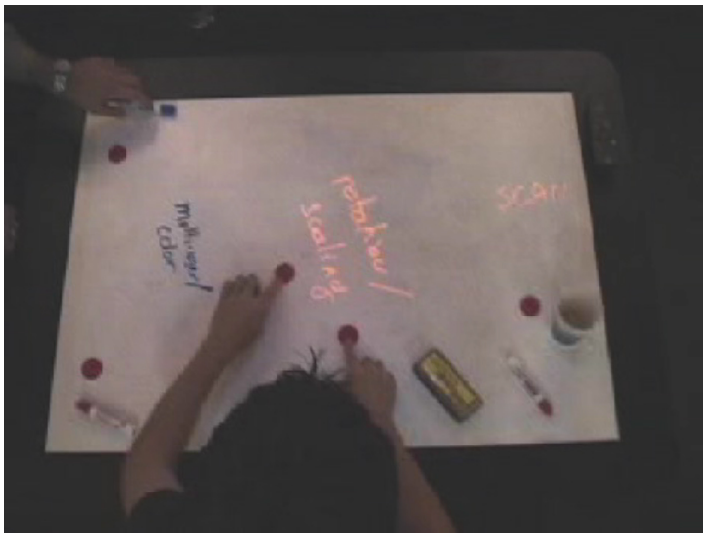

Figure 5. *La Table Magique. Deux jetons sont couplés par mise en contact. Dès lors, ils permettent de sélectionner des inscriptions numériques ou écrites au feutre en les éloignant l'un de l'autre, puis, comme le montre l'image, d'appliquer aux inscriptions sélectionnées à la fois des rotations et des changements de taille. Le couplage prend fin lorsque l'un des jetons est masqué par la main.*

Avec le PDS (Portable Projection Screen), toute surface, une fois couplée au SCP (Steerable Camera-Projector) devient interactive (Borkowski *et al.*, 2003). Comme le montre la Figure 6, le SCP est l'assemblage d'une caméra et d'un vidéoprojecteur asservis à un même moteur. Ce moteur, piloté par le système, permet de projeter de l'information numérique sur une surface, par exemple, le morceau de carton tenu dans les mains de la Figure 7. La caméra alimente un système de vision par ordinateur qui suit les mouvements du carton en sorte de maintenir la projection de l'information sur le carton. Le système de vision détecte également la présence de doigts sur des widgets projetés, conférant au morceau de carton des propriétés d'interactivité (Cf. Figure 8).

Figure 6. *Le SCP (Steerable Camera-Projector)*

Figure 7. *Un morceau de carton tenu dans la main. À droite, le morceau de carton une fois couplé au SCP.*

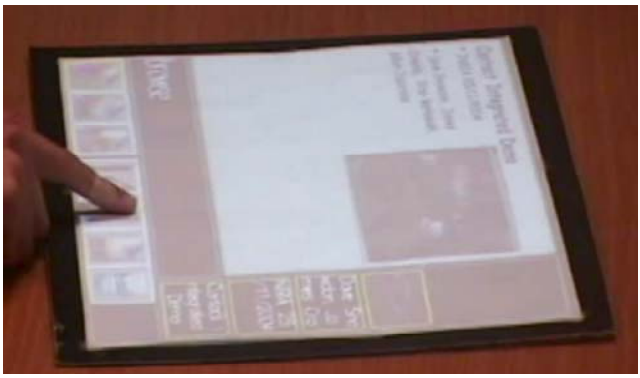

Figure 8. *Un utilisateur « cliquant » un widget au moyen de son doigt*

Jusqu'ici, nous avons présenté quelques exemples représentatifs de la littérature (il y en a bien d'autres), mais de manière informelle sans nous interroger sur la nature du couplage. Ces exemples nous permettront également d'illustrer certains de nos exemples. Nous présentons maintenant notre cadre analytique.

3 Cadre analytique

Nous nous plaçons dans le cadre de couplages dynamiques répondant à l'exigence suivante : l'utilisateur doit disposer à tout instant des "indices nécessaires et suffisants" pour juger de l'existence ou de la possibilité de couplage entre deux ressources d'interaction. Cette exigence tient de la nécessité pour l'utilisateur de construire, au cours du processus d'interaction, une représentation mentale en conformité avec l'état interne du système (Norman et Draper, 1986). Cette exigence précisée, nous posons que le couplage de ressources d'interaction passe par une suite d'états observables et contrôlables par l'utilisateur. Tout état se définit au moyen de critères. Dans ce qui suit, nous présentons notre définition de la notion de couplage, puis nous introduisons les critères caractéristiques d'un état de couplage et les conditions de transition entre les états. Ainsi, un couplage obéit à un cycle de vie.

3.1 Couplage de ressources d'interaction : définition

Le couplage de deux ressources d'interaction est l'action de lier ces ressources de manière à ce qu'elles opèrent conjointement pour fournir un ensemble de fonctions que, seules, elles ne pourraient pas fournir (Barralon *et al.*, 2004). En reprenant l'exemple de la Figure 3, les deux écrans sont couplés pour obtenir l'extension de la surface d'affichage ; ils pourraient également être couplés pour obtenir une fonction de duplication de l'affichage d'un des deux écrans. Dans ce

dernier cas, même si l'espace d'affichage est identique (la même information est affichée deux fois) le fait de pouvoir y accéder par le biais de deux ressources d'interaction différentes, justifie le fait de considérer qu'il s'agit d'une nouvelle fonction.

3.2 Un état : trois critères de vérité

Soient ***R*** l'ensemble des ressources d'interaction, ***F*** l'ensemble des fonctions de couplage et ***C*** l'ensemble des couplages.

Soient r_1 et r_2 deux ressources d'interaction, F l'ensemble des fonctions obtenues par le couplage c de r_1 avec r_2, noté (r_1, c, r_2) ; avec r_1 et $r_2 \in \boldsymbol{R}$, $F \subset \boldsymbol{F}$ et $c \subset \boldsymbol{C}$.

Un état de (r_1, c, r_2) est défini par l'ensemble des trois critères de vérité suivants :

- r_1 est (couplée /non couplée) avec r_2 pour c,
- r_1 est (verrouillée /non verrouillée) pour r_2 et pour c,
- r_1 est (couplable /non couplable) à r_2 pour c, ou bien r_1 est (découplable/non découplable) de r_2 pour c.

Ou, si l'on utilise les prédicats équivalents :

- *Couplée* (r_1, c, r_2) est vrai si et seulement si $F \neq \varnothing$. Si $F=\varnothing$, *Couplée* (r_1, c, r_2) est faux et *NonCouplée* (r_1, c, r_2) est vrai.
- *Verrouillée* (r_1, c, r_2) est vrai si l'état de r_1 ne permet pas de changer l'état de (r_1, c, r_2). Ce prédicat traduit l'indisponibilité de r_1 pour entrer ou sortir du couplage c avec r_2. Par exemple, un utilisateur ne veut pas joindre à l'écran public, l'écran privé de son PDA utilisé à des fins personnelles. L'état de (r_1, c, r_2) est contraint au statu quo jusqu'à ce que *Verrouillée* (r_1, c, r_2) devienne faux ou, de manière duale *NonVerrouillée* (r_1, c, r_2) devienne vrai.
- *Couplable* (r_1, c, r_2) est une expression de prédicats autres que *Couplée* (r_1, c, r_2) et *Verrouillée* $(r1, c, r2)$. Elle regroupe toutes les conditions nécessaires à la réalisation de (r_1, c, r_2), autres que les conditions sur l'existence de F ou sur le verrouillage de r_1 vis-à-vis de r_2 dans c. Par exemple, la compatibilité de forme et de rôle, peuvent être modélisées par *Couplable*. De manière symétrique, *Découplable* exprime toutes les conditions, autres que les conditions sur l'existence de F ou sur le verrouillage, nécessaires à la réalisation de la destruction de (r_1, c, r_2).

3.3 Transitions

Les transitions entre états correspondent à des événements dont la prise en compte par le système modifie la valeur de l'un des critères. Ces événements traduisent soit la demande de couplage/découplage (c'est-à-dire la demande de lancement ou de suspension des fonctions de F), soit le verrouillage/déverrouillage de ressources, soit leur couplabilité/découplabilité.

3.4 Cycle de vie

L'automate de la Figure 9 modélise le cycle de vie de (r_1, c, r_2). Un automate similaire modélise celui de (r_1, c, r_2).

Le cycle de vie comprend deux sous-automates, l'un formé par les états 1, 2, 3, 4 pour lesquels *Couplée* (r_1, c, r_2) est vrai, l'autre par les états 5, 6, 7, 8 pour lesquels *Couplée* (r_1, c, r_2) est faux, les états 4 et 6 servant de passerelle entre les deux sous-automates. L'état 4 correspond à la situation où toutes les conditions sont réunies pour que (r_1, c, r_2) soit réalisé. Il ne manque qu'un événement de demande de couplage pour entrer dans l'état 6.

À titre d'illustration, déroulons l'automate pour le cas de deux jetons j_1 et j_2 de la Table Magique. A l'initialisation du système, (j_1, c, j_2) est dans l'état 4 : j_1 est ni couplé ni verrouillé et couplable vis-à-vis de j_2 puisqu'il est sur la table, qu'il est de même couleur, de même forme et de même taille que j_2 (le suivi des jetons s'appuie sur un système de vision par ordinateur fondé sur un modèle de couleur). La mise en contact de j_1 et j_2 signifie, pour le système, une demande de couplage. (j_1, c, j_2) entre dans l'état 6 et la fonction "sélectionner un patch" est disponible. Si l'utilisateur sélectionne un patch, puis masque j_2 de la main, (j_1, c, j_2) revient en 4. Mais comme j_1 est maintenant attaché à un patch (couplage jeton-table), j_1 est verrouillé pour (j_1, c, j_2), et (j_1, c, j_2) passe de l'état 4 à l'état 3. (j_1, c, j_2) reviendra à l'état 4 dès que j_1 sera détaché de la table, c'est-à-dire dès qu'il sera masqué par la main. S'il est verrouillé pour (j_1, c, j_2), j_1 n'est pas verrouillé pour (j_1, c', j_2) où c' donne accès aux fonctions de rotation, de zoom et de destruction de patch. (j_1, c', j_2) est dans l'état 4. Il passe en 6 dès que l'utilisateur pose j_2 à l'intérieur du patch auquel j_1 est attaché. L'analyse de ces quelques exemples de couplage montre combien cette opération, simple de principe, soulève de nombreux problèmes potentiels d'utilisabilité. Nous les analysons plus avant dans la section qui suit.

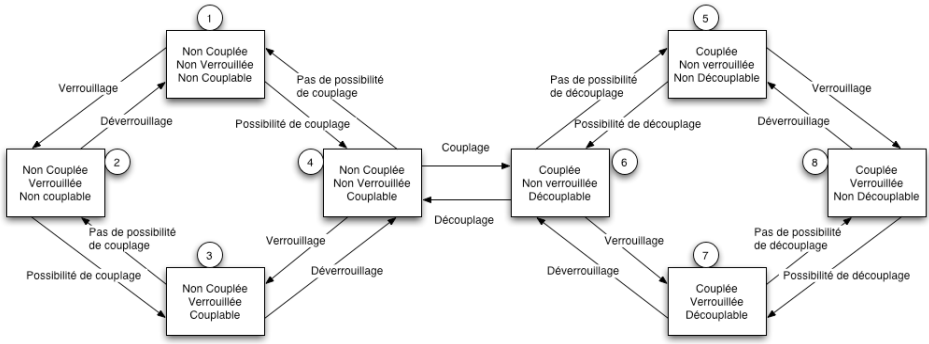

Figure 9. *Automate du cycle de vie du couplage (r_1, c, r_2). Par souci de clarté du schéma, la fermeture transitive entre états n'est pas représentée, mais les états 1 et 3, 2 et 4, 5 et 7, 6 et 8 sont évidemment reliés.*

3.5 Propriétés ergonomiques

Nous analysons ci-dessous l'utilisabilité du couplage selon les deux grandes dimensions, souplesse et robustesse de l'interaction introduites dans (Gram et Cockton, 1996). D'autres cadres d'analyse comme les règles de Bastien et Scapin (1993) ou de Bellotti (Belloti *et al.*, 2002) sont applicables. Ainsi le choix des propriétés de Gram et Cockton est arbitraire mais permet de mettre en évidence les « nouveaux » problèmes liés au couplage au regard des propriétés communément admises en Interaction Homme-Machine.

Souplesse de l'interaction

Atteignabilité : "Capacité du système à permettre à l'utilisateur de naviguer dans l'ensemble des états observables du système". Appliquée à notre problème, l'atteignabilité signifie que l'utilisateur doit pouvoir atteindre les états (couplé, verrouillé, couplable) et leur inverse (découplé, déverrouillé, découplable). Il convient donc que l'utilisateur dispose de techniques d'interaction pour atteindre chacun de ces états. Par exemple, pour le verrouillage, Hinckley (2003) propose de masquer avec la main la tablette dont on ne veut pas partager le contenu.

Non-préemption : "Le prochain but souhaité par l'utilisateur est directement atteignable". Pour notre problème, on retiendra que les états souhaités par l'utilisateur doivent être accessibles à tout instant et ceci par des trajectoires d'interaction optimisées. C'est le cas des ConnecTables (Tandler *et al.*, 2001) qui peuvent toujours être couplées ou découplées en un seul geste.

Interaction multifilaire : "Capacité du système à permettre la réalisation de plusieurs tâches". Dans le contexte de notre étude, cette propriété traduit la capacité du système, comme la Table Magique (Bérard, 2003), à traiter plusieurs couplages simultanément ; ou bien, comme cela peut se produire en interaction proximale, plusieurs utilisateurs demandent en même temps le couplage d'entités avec une même entité cible. Soit le système gère l'accès concurrent et l'exprime, soit les conflits sont résolus de manière sociale.

Adaptabilité : "Personnalisation du système sur intervention explicite de l'utilisateur". Aucun exemple de l'état de l'art sur le couplage de ressources d'interaction, n'illustre cette propriété. Couplage/Découplage, Verrouillage/Déverrouillage, etc., sont conventionnels et immuables.

Migrabilité de tâche : "Capacité de délégation dynamique de tâches entre le système et l'utilisateur ou entre utilisateurs. C'est un changement dynamique de l'acteur(s) responsable(s) de l'accomplissement de la tâche".
Au regard du couplage, cette propriété s'interprète comme la possibilité de déléguer une partie du couplage au système. Par exemple, les claviers et souris d'une même plate-forme sont automatiquement couplés dès le lancement du système.

Robustesse de l'interaction
Observabilité : "Capacité du système à rendre perceptible l'état pertinent du système". Cette propriété, appliquée à l'automate de couplage, impose de rendre observable l'ensemble des états de l'automate. En particulier, nous avons vu dans l'analyse de l'état de l'art que les états « couplable » et « non couplable » ne sont généralement pas observables.

Honnêteté : "Capacité du système à rendre observable l'état du système sous une forme conforme à cet état et qui engendre une interprétation correcte de la part de l'utilisateur". Autrement dit, l'état du couplage doit non seulement être observable à tout instant, mais ce rendu doit être compris correctement par l'utilisateur. Dans I-AM où l'utilisateur peut disposer de plusieurs souris et claviers couplables et découplables à volonté, l'évaluation de l'état actuel des couplages peut poser problème : l'utilisateur est amené à se demander quel clavier est couplé avec quelle souris. Leur proximité spatiale sur la surface de travail ne suffit pas à lever l'ambiguité.

Curabilité : "Capacité pour l'utilisateur de corriger une situation non désirée". Dans l'automate du couplage, cette propriété se traduit par la possibilité de découpler et de déverrouiller à tout instant. La curabilité impose une vigilance toute particulière sur les conséquences du découplage. Par exemple, considérons un écran e_1 couplé à une souris et à un écran e_2 en sorte que le curseur de la souris couvre l'ensemble des deux écrans. Si l'on découple e_1 et e_2, la souris reste-t-elle couplée à e_1 ou à e_2 ?

Prévisibilité : "Capacité pour l'utilisateur de prévoir, pour un état donné, l'effet d'une action". Ici, l'utilisateur doit anticiper les conséquences d'un couplage ou d'un découplage. Chez Hinckley, selon le geste, le couplage de deux tablettes ne fournit pas le même résultat et ce résultat, pour un utilisateur néophyte, n'est pas prévisible. Ou encore, soient deux souris couplées s_1 et s_2, et deux claviers couplés c_1 et c_2. Que se passe-t-il si l'utilisateur couple s_1 à c_1 ? Le couplage est-il transitif ? Si tel est le cas, s_1 devient également couplée à c_2 ; de même, s_2 est couplée à c_1 et c_2. Ou encore, si s_1 est couplée à un écran e_1, et s_2 à un écran e_2. Quels sont les effets de bord pour les souris suite au couplage de e_1 et e_2 ? Pour l'utilisateur, ces conséquences sont-elles prévisibles ?

Synthèse

L'analyse des propriétés de souplesse et de robustesse de l'interaction appliquées à l'automate du couplage de ressources d'interaction, montre, avant toute implémentation, qu'il s'agit d'un problème plus complexe qu'il n'y paraît. Il montre aussi que l'état de l'art s'est peu préoccupé du problème ou s'y est attaché de manière plutôt exploratoire que rationnelle. Dans la section qui suit, nous formalisons le couplage de ressources d'interaction en vue de répondre aux questions laissées sans réponse, notamment sur la transitivité du couplage.

4 Formalisation

Nous proposons deux notations pour le couplage de ressources d'interaction : l'une sous forme de graphes, faciles à lire ; l'autre sous forme algébrique plus concise pour exprimer les conséquences d'un couplage de ressources sur l'ensemble des couplages existants. Prenons un petit exemple tiré de la métaphore du bureau qui nous est familière.

4.1 Un exemple

En termes de couplage, la métaphore du bureau se traduit de la manière suivante : un écran $écran_1$ est couplé à une souris $souris_1$ ($écran_1$, $souris_1$, c_1). Cet écran est également couplé à un clavier $clavier_1$ ($écran_1$, $clavier_1$, c_2). Enfin, le clavier et la souris sont couplés ($clavier_1$, $souris_1$, c_3) de telle manière que le clavier offre la fonction « modifiers »[1] à la souris et que cette dernière fournit la fonction « focus »[2] au clavier.

Étendons maintenant l'exemple : un second écran $écran_2$ et une seconde souris $souris_2$ sont couplés ($écran_2$, $souris_2$, c_4) pour la fonction de désignation. On nomme *configuration initiale* l'ensemble des couplages $\{c_1, c_2, c_3, c_4\}$. Supposons que l'utilisateur de cette configuration initiale demande le couplage de $écran_1$ et $écran_2$ en sorte que $écran_1$ étende la surface d'affichage d'$écran_2$ et réciproquement ($écran_1$, $écran_2$, c_5). On dit que c_5 est un *couplage initiateur*.

La réalisation de c_5 appelle plusieurs comportements possibles :

1. Aucun couplage supplémentaire n'est créé,
2. L'utilisateur conserve les possibilités de saisie de texte dans son espace d'affichage,
3. Tout dispositif d'entrée doit pouvoir agir sur n'importe quel écran.

[1] La fonction « modifiers » fournit des états supplémentaires aux événements souris en évaluant si les touches du clavier « shift » et « control » (par exemple) sont enfoncées lors d'un événement souris.

[2] Les événements clavier sont envoyés aux interacteurs ayant reçu le focus via la souris. Sans cette fonction, le clavier n'a pas de fonction de navigation entre les interacteurs hormis par la touche « tabulation ».

Développons les points 2 et 3 qui nécessitent la génération de nouveaux couplages.

Dans le cas 2, le système doit générer les couplages (*écran*$_2$, *clavier*$_1$, c_6) pour que la saisie de texte puisse être effectuée sur *écran*$_2$ mais également (*clavier*$_1$, *souris*$_2$, c_7) pour que l'utilisateur puisse désigner les interacteurs qui doivent recevoir les événements clavier. On appelle cet ensemble de couplages *configuration terminale* puisque présentant les « couplages terminaux » à la fin de la phase de génération du système. La Figure 10 montre la configuration terminale correspondant au cas 2. En pratique, l'utilisateur peut manipuler de l'information au moyen de *souris*$_1$ et de *clavier*$_1$ sur *écran*$_1$. Il peut faire migrer une fenêtre d'*écran*$_1$ vers *écran*$_2$ sans pour autant que *souris*$_1$ puisse « sortir » de *écran*$_1$. À partir du moment où un interacteur se retrouve sur *écran*$_2$ (même s'il avait le focus fourni par *souris*$_1$) alors il est nécessaire de redonner le focus avec *souris*$_2$ pour saisir du texte. Le comportement est similaire au regard d'*écran*$_2$, *souris*$_2$ et *clavier*$_1$.

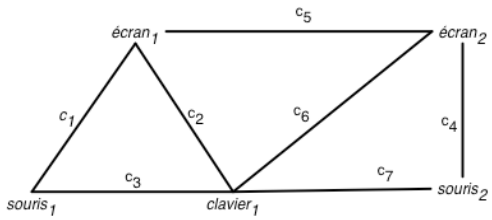

Figure 10. *Configuration terminale du cas 2*

La Figure 11 présente la configuration terminale obtenue pour le cas 3. Les couplages (*souris*$_1$, *écran*$_2$, c_7) et (*clavier*$_1$, *écran*$_2$, c_6) sont générés : puisque l'affichage de *écran*$_1$ est étendu sur *écran*$_2$, il est « normal » que *souris*$_1$ et *clavier*$_1$ accèdent à l'information de la même manière que sur *ecran*$_1$. Pour les mêmes raisons, (*souris*$_2$, *écran*$_1$, c_8) est généré. Le comportement des interfaces graphiques est moins compliqué que pour le cas 2. En effet, pour chaque dispositif d'entrée, *écran*$_1$ et *écran*$_2$ sont considérés comme formant un seul écran. De ce fait, chaque dispositif d'entrée a la même fonction quel que soit l'écran.

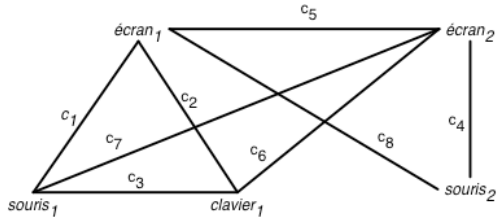

Figure 11. *Configuration terminale pour le cas 3*

Ces exemples posent évidemment le problème de gestion de focus multiples sur une application ou de désambiguïsation de focus puisque deux focus peuvent être définis simultanément. On ne traite pas plus avant ce type d'implications.

4.2 Notation graphique

La représentation des couplages est réalisée sous forme de graphe non orienté, dont les sommets désignent les ressources d'interaction et les arêtes les couplages entre deux ressources. L'ensemble des sommets est noté S, l'ensemble des arêtes A. Les arêtes portent également le nom du couplage qu'elles représentent (c_1, c_2, …, c_n)

sous la forme d'étiquettes. Un couplage initiateur est dénoté par le symbole « * », les couplages déduits par « = » et les couplages évalués par « ? ». Les couplages déduits sont des couplages évalués retenus dans le processus de génération des couplages suite à la survenue d'un couplage initiateur. Les couplages initiateurs, déduits et évalués sont représentés en trait pointillé pour mettre en évidence leur caractère particulier, voire transitoire. La Figure 12 correspond au cas 2 de l'exemple. Les couplages c_1, c_2, c_3 et c_4 constituent la configuration initiale à laquelle s'ajoutent c_5, le couplage initiateur (celui demandé par l'utilisateur), les couplages évalués c'_6, c'_4 et les deux couplages déduits c_6 et c_7.

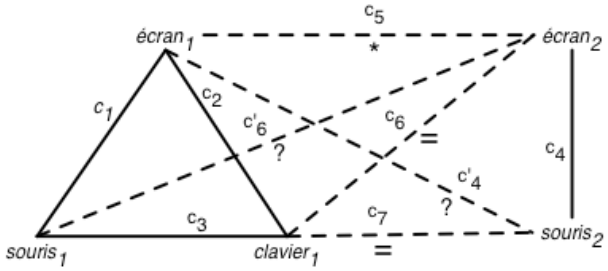

Figure 12. *Notation graphique représentant la configuration initiale, le couplage initiateur, les couplages évalués et les couplages déduits du cas 2 de l'exemple*

Soient les prédicats :

- $r_1(c)$ (resp. $r_2(c)$) renvoie la première (resp. la seconde) ressource d'interaction de c.
- $F(r_1, r_2)$ retourne la fonction obtenue pas le couplage de r_1 à r_2.
- *Compatible*(f_1, f_2, f_3) renvoie vrai si les fonctions f_1 et f_2 autorisent la génération de f_3. Cette compatibilité est décidée par conception. Elle est équivalente à la couplabilité de deux ressources introduite dans la section 3.2.

La génération des couplages déduits suit le schéma suivant :

1. Pour chaque couplage initiateur, on considère les chaînes de longueur 2 contenant $r_1(c)$ et $r_2(c)$.
2. Les chaînes obtenues sont de *type* $r_1(c)$–$r_2(c)$–s ou s–$r_1(c)$ –$r_2(c)$
3. Si s–$r_1(c)$–$r_2(c)$ et *Compatible*$(F(c), F(r_1(c),s), F(r_2(c),s))$ alors ajouter l'arête $r_2(c)$–s
   Si $r_1(c)$–$r_2(c)$–s et *Compatible*$(F(c), F(r_2(c),s), F(r_1(c),s))$ alors ajouter l'arête $r_1(c)$-s

Ou

```
Pour tout couplage initiateur c
   Construire l'ensemble Sc des sommets tel que
      s∈Sc ⇔  s∈chaîne ∧ longueur(chaîne)= 2
              ∧ r1(c)∈chaîne ∧ r2(c) ∈chaîne
   pour tout s de Sc
      si s≠r1(c) et s≠r2(c)
         si arête(s,r1(c)) ∈ A
            si compatible(F(c), F(s, r1(c)), F(s, r2(c)))
               A = A ∪ new arête(r2(c), s)
         sinon
            si compatible(F(c), F(s, r2(c)), F(s, r1(c)))
               A = A ∪ new arête(r1(c), s)
```

De façon générale, on étudie chaque arête permettant la fermeture transitive des chaînes de longueur 2 contenant $r_1(c)$ et $r_2(c)$. Afin d'obtenir une génération complète, chaque nouvelle arête créée (nouveau couplage) prend le statut de couplage initiateur, donnant lieu à une nouvelle génération.

4.2.1 Application à l'exemple : cas 2

La Figure 13 montre la première génération où les chaînes prises en compte sont c_1–c_5 et c_2–c_5 et les couplages évalués c'_6 et c_6 ainsi que la chaîne c_5–c_4 pour évaluer c'_4.

De ces évaluations, seul c_6 sera conservé car, par conception, les fonctions de c_2 et c_5 sont compatibles pour la production de la fonction de c_6. En effet, c_5 étend la surface d'affichage d'*ecran*$_1$ sur *écran*$_2$. Aussi, puisque que l'on conserve les capacités de saisie de texte, *clavier*$_1$ doit être couplé avec tous les écrans étendant *écran*$_1$. c_6 constitue le couplage initiateur de la génération suivante.

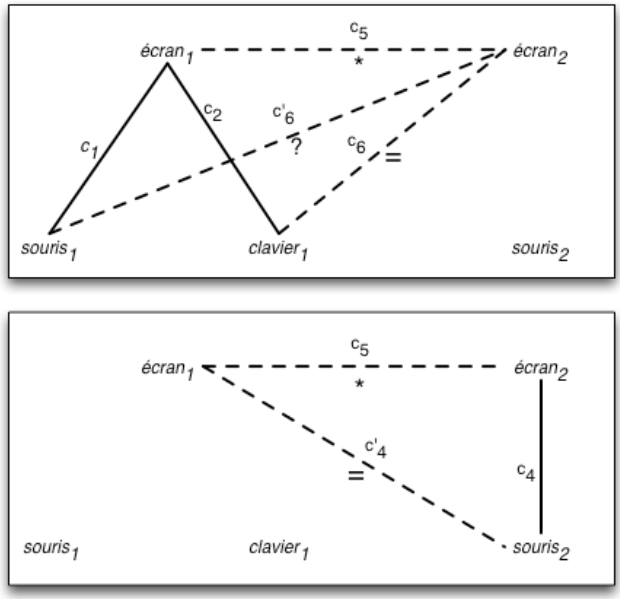

Figure 13. *Couplages évalués pour les chaînes de type s-écran*$_1$*-écran*$_2$ *en haut, écran*$_1$*-écran*$_2$*-s en bas*

La Figure 14 montre la deuxième étape de génération pour laquelle c_6 est maintenant considéré comme un couplage initiateur et duquel est déduit le couplage c_7. Si un clavier est couplé à un écran, mais pas à une souris, il peut certes envoyer des événements mais ne peut saisir du texte n'importe où à l'écran. c_6 et c_4 sont compatibles pour la génération de c_7.

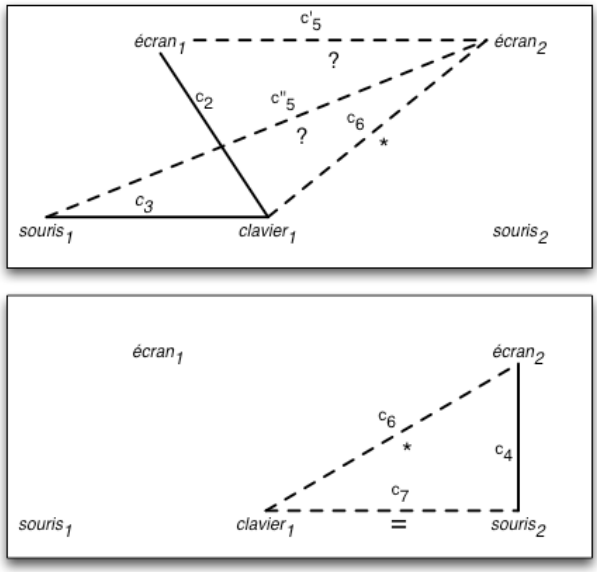

Figure 14. *Couplages évalués pour les chaînes de type s-clavier$_1$-écran$_2$ en haut, clavier$_1$-écran$_2$-s en bas*

4.2.2 Application à l'exemple : cas 3

La Figure 15 montre les couplages évalués et déduits pour le cas 3 de notre exemple. Tous les couplages évalués dans la première génération sont acceptés. La deuxième évaluation (avec c_6, c_7 et c_8 en couplage initiateur) ne produit rien.

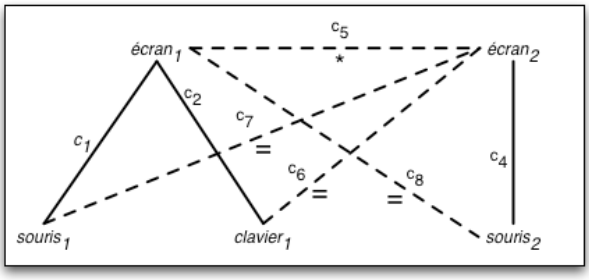

Figure 15. *Couplages évalués pour le couplage initiateur c_5*

La notation graphique présente l'inconvénient d'être lourde à mettre en œuvre pour représenter le processus de génération. La notation mathématique qui suit permet d'exprimer le processus de manière concise.

4.3 Formalisation mathématique

4.3.1 Description

La formalisation est régie par deux opérateurs « * » appelé opérateur de génération et « + » l'opérateur d'union. L'état du système en termes de couplage est noté $[c_1 + c_2 + \ldots + c_n]$ représentant l'union de c_1, c_2, … *et* c_n. L'opérateur « * » est

distributif sur « + » et la priorité relative de ces deux opérateurs est également respectée.

L'ajout d'un couplage se déroule en deux phases. Dans un premier temps, le couplage initiateur est créé et ajouté à la liste des couplages composant le système, puis on évalue si d'autres couplages doivent être déduits.

La première phase est prise en charge par la *règle d'ajout* :

$c_p + [c_1 + c_2 + \ldots + c_n]$ qui produit $[c_p + c_1 + c_2 + \ldots + c_n]$

La deuxième phase est réalisée à l'aide de la *règle de génération :*

$c_p * [c_p + c_1 + c_2 + \ldots + c_n] = c_p * c_p + c_p * c_1 + c_p * c_2 + \ldots + c_p * c_n$

La résolution de $c_p * c_p + c_p * c_1 + c_p * c_2 + \ldots + c_p * c_n$ nécessite d'évaluer un par un les termes $c_p * c_i$. On peut d'ores et déjà noter que, par convention, le terme $c_p * c_p$ ne peut rien produire. Afin de mieux comprendre l'évaluation d'un terme, on procède à la réécriture de chaque couplage sous la forme simplifiée *(R, R, F)* :

$c_p * c_i = (r_{p1}, r_{p2}, f_p) * (r_{i1}, r_{i2}, f_i)$

Deux cas sont alors possibles : les deux couplages sont soit transitifs, soit intransitifs :

c_p et c_i sont transitifs ssi $(r_{p1}=r_{i1} \vee r_{p1}=r_{i2} \vee r_{p2}=r_{i1} \vee r_{p2}=r_{i2}) \wedge Compatible(f_p, f_i, f) \wedge c_p \neq c_i$

c_p et c_i sont intransitifs sinon.

Si c_1 et c_2 sont transitifs alors :

$c_p * c_i = (r_{p1}, r_{p2}, f_p) * (r_{i1}, r_{i2}, f_i) = c_{res}$ avec

Compatible $(f_p, f_i f')$

$c_{res} = (r_{p2}, r_{i2}, f')$ si $r_{p1}=r_{i1}$

$c_{res} = (r_{p2}, r_{i1}, f')$ si $r_{p1}=r_{i2}$

$c_{res} = (r_{p1}, r_{i2}, f')$ si $r_{p2}=r_{i1}$

$c_{res} = (r_{p1}, r_{i1}, f')$ si $r_{p2}=r_{i2}$

$Transitif(c_p, c_i) = vrai$

Si c_1 et c_2 sont intransitifs alors :

$c_p * c_i = (r_{p1}, r_{p2}, f_p) * (r_{i1}, r_{i2}, f_i) = \varnothing$

$Transitif(c_p, c_i) = faux$

On voit apparaître la correspondance avec la notation sous forme de graphes. Un nouveau couplage c_{res} n'est créé que si c_p et c_i sont transitifs, c'est-à-dire s'ils possèdent une ressource en commun. Cette condition correspond aux chaînes de longueurs 2 contenant $r_1(c_p)$ et $r_2(c_p)$

Dans le cas de deux couplages transitifs c_p et c_i, la fonction du couplage c_{res} produit se doit d'être compatible avec $Fc(c_p)$ et $Fc(c_i)$, tout comme dans la notation graphique. La règle de production telle que l'on vient de la décrire permet de générer de nouveaux couplages. La résolution complète de cette règle suppose que chaque couplage généré donne lieu à la ré-application de la règle de génération. Autrement dit, chaque couplage c_{res} donne lieu à l'application de la règle d'ajout puis celle de génération.

4.3.2 Application à l'exemple cas 2

Configuration initiale : $[c_1 + c_2 + c_3 + c_4]$
Couplage initiateur : c_5

Règle d'ajout :
$c_5 + [c_1 + c_2 + c_3 + c_4] = [c_5 + c_1 + c_2 + c_3 + c_4]$

Règle de génération :
$c_5 * [c_5 + c_1 + c_2 + c_3 + c_4] = c_5 * c_5 + c_5 * c_1 + c_5 * c_2 + c_5 * c_3 + c_5 * c_4$
$= \varnothing + c_5 * c_1 + c_5 * c_2 + c_5 * c_3 + c_5 * c_4$
$= \varnothing + \varnothing + c_6 + \varnothing + \varnothing$

Règle d'ajout :
$c_6 + [c_5 + c_1 + c_2 + c_3 + c_4] = [c_6 + c_5 + c_1 + c_2 + c_3 + c_4]$

Règle de génération :
$c_6 * [c_6 + c_5 + c_1 + c_2 + c_3 + c_4] = c_6 * c_6 + c_6 * c_5 + c_6 * c_1 + c_6 * c_2 + c_6 * c_3 + c_6 * c_4$
$= \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + c_7$

Règle d'ajout :
$c_7 + [c_6 + c_5 + c_1 + c_2 + c_3 + c_4] = [c_7 + c_6 + c_5 + c_1 + c_2 + c_3 + c_4]$

Règle de génération :
$c_7 * [c_7 + c_6 + c_5 + c_1 + c_2 + c_3 + c_4] = c_7*c_6 + c_7*c_6 + c_7*c_5 + c_7*c_1 + c_7*c_2 + c_7*c_3 + c_7*c_4$
$= \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$

Configuration terminale: $[c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7]$

4.3.3 Application à l'exemple cas 3

Configuration initiale : $[c_1 + c_2 + c_3 + c_4]$
Couplage initiateur : c_5

Règle d'ajout :
$c_5 + [c_1 + c_2 + c_3 + c_4] = [c_5 + c_1 + c_2 + c_3 + c_4]$

Règle de génération :
$c_5 * [c_5 + c_1 + c_2 + c_3 + c_4] = c_5 * c_5 + c_5 * c_1 + c_5 * c_2 + c_5 * c_3 + c_5 * c_4$
$= \varnothing + c_6 + c_7 + \varnothing + c_8$

Règle d'ajout :
$c_6 + c_7 + c_8\ [c_5 + c_1 + c_2 + c_3 + c_4] = [c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8]$

Règle de génération :
$(c_6 + c_7 + c_8) * [c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8]$
Pour plus de lisibilité, on scinde le calcul en trois

1. $c_6*c_1+c_6*c_2+c_6*c_3+c_6*c_4+c_6*c_5+c_6*c_6+c_6*c_7+c_6*c_8$
2. $+c_7*c_1+c_7*c_2+c_7*c_3+c_7*c_4+c_7*c_5+c_7*c_6+c_7*c_7+c_7*c_8$
3. $+c_8*c_1+c_8*c_2+c_8*c_3+c_8*c_4+c_8*c_5+c_8*c_6+c_8*c_7+c_8*c_8$

1. $\varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$
2. $+ \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$
3. $+ \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$

Configuration terminale: $[c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8]$

Afin d'alléger l'écriture, la règle de génération peut ne pas comporter tous les couplages du système en partie droite, mais uniquement les couplages transitifs. En ce sens, l'écriture des couplages sous la forme (r_1, r_2, f_1) facilite l'élimination des

couplages non transitifs. Ce premier filtre permet de n'évaluer que les couplages compatibles.

Nous avons vu que la transitivité entre couplages s'appuie sur la notion de compatibilité de fonctions. Cette compatibilité est établie à la conception du système. Jusqu'ici, nous avons raisonné de manière théorique sans nous attarder sur la nature des fonctions de couplage et leur effet sur la compatibilité. Nous pensons que les fonctions génériques et les heuristiques ont un rôle à jouer.

4.4 Fonctions génériques et heuristiques

Dans la métaphore du bureau, les fonctions copy-paste-undo-redo sont génériques. Par analogie, il devrait exister des fonctions de couplage génériques. Nous pensons à la notion de fusion et son inverse, la fission. Le couplage de deux ressources est une fonction de fusion si les services fournis par chacune de ces ressources prises séparément sont conservés et si chacune de ces ressources peut être utilisée indifféremment (équivalence fonctionnelle au sens de CARE (Coutaz *et al.*, 1995).

Synchronous Gesture (Hinckley, 2003) offre un exemple de fusion entre écrans : quant un écran annexe un second écran, l'utilisateur obtient un espace d'affichage unique, union des espaces d'affichage des écrans originaux. Un pointeur souris manipulable par deux souris est un autre exemple de fusion : les deux souris sont associées au même pointeur. L'utilisateur peut employer l'une ou l'autre indifféremment pour désigner un point de l'écran. L'analyse de l'état de l'art, et l'expérience pratique tendent à montrer que la fonction de fusion est compatible avec toute autre fonction de couplage.

L'expérience conduira à l'émergence d'heuristiques et de bons principes. Pour l'instant, le domaine se construit et se contente de proposer des techniques d'interaction.

5 Technique d'interaction pour le couplage

Selon Foley, une technique d'interaction est une manière d'utiliser des dispositifs d'entrée pour saisir de l'information (« interaction techniques are ways to use input devices to enter information into the computer ») (Foley *et al.*, 1990). Selon Bowman, une technique d'interaction est « une méthode selon laquelle l'utilisateur réalise une tâche sur un ordinateur via l'interface utilisateur » (Bowman et Hodges, 1999). Pour notre part, une technique d'interaction est un langage dont le vocabulaire et les éléments syntaxiques sont produits au moyen de ressources d'interaction. La sémantique d'une phrase de ce langage correspond à une fonction ou service du système. Ici, nous nous intéressons aux techniques d'interaction mise à la disposition de l'utilisateur pour spécifier un couplage de ressources d'interaction. La sémantique de la phrase de couplage correspond à la fonction de couplage.

5.1 Désignation de ressources d'interaction

Dans les techniques précédemment citées, certaines comme Synchronous Gesture ou les ConnectTables sont très ancrées dans le physique, d'autres comme le Clic&Couple ou un configurateur (Barralon *et al.*, 2004) sont résolument logicielles. Enfin comme dans les DataTiles, l'utilisateur manie un instrument pour désigner les ressources d'interaction. Afin de classifier ces techniques sur le plan de la désignation des ressources d'interaction, nous reprenons une notation graphique proposée par Renevier (2004) concernant les systèmes mixtes. Cette notation, dont on va faire un bref rappel, offre l'avantage de modéliser aussi bien des entités physiques que des

entités numériques. Nous utiliserons la notion d'instrument telle qu'elle a été décrite par Lachenal (2004). Une entité physique possède des propriétés intrinsèques et va jouer (ou non) le rôle d'instrument pour l'interaction. Cette propriété n'est donc pas figée pour une entité physique, mais c'est réellement l'utilisateur (ou le système) qui va lui conférer le rôle d'instrument. Par exemple, les jetons de la Table Magique (Bérard, 2003) ne sont pas intrinsèquement des instruments mais lorsque le système est en route et que la caméra peut les suivre, les jetons jouent alors le rôle d'instruments.

5.1.1 Rappel de la notation

La Figure 16 montre l'ensemble des entités de la notation de Renevier. Pour synthétiser :

- Un rond (respectivement un carré) plein correspond à un objet (respectivement un instrument) physique.
- Un rond (respectivement un carré) hachuré correspond à un objet (respectivement un instrument) numérique.
- Un rond (respectivement un carré) plein ou hachuré entouré par un autre rond (respectivement un autre carré) plein ou hachuré correspond à une représentation.

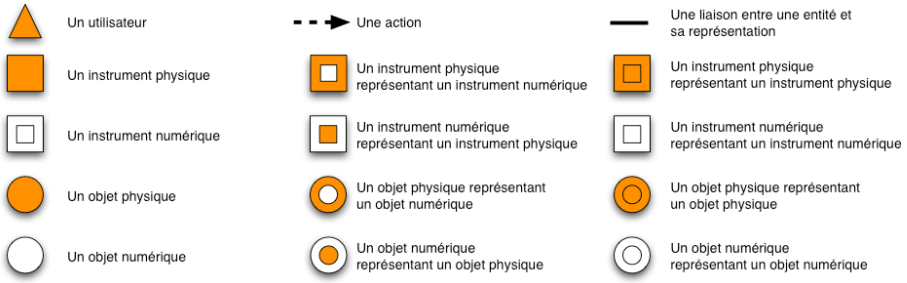

Figure 16. *Récapitulatif des types d'entités et des relations*

En guise d'illustration, la Figure 17 montre l'exemple d'une interaction classique. Un utilisateur manipule le curseur souris (instrument numérique représentant un instrument physique) grâce à une souris (instrument physique) et va agir sur le window manager (représentation numérique de la surface) d'un écran (objet physique).

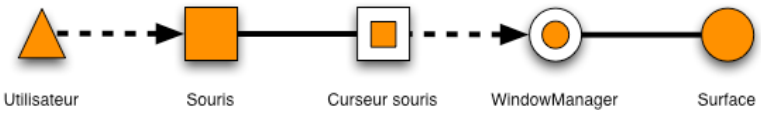

Figure 17. *Exemple d'interaction entre une souris manipulée par un utilisateur et une surface.*

5.1.2 Application aux techniques d'interaction pour le couplage

L'analyse de l'état de l'art fait ressortir quatre grandes classes de techniques (Figure 18) que nous détaillons en ajoutant une analyse des avantages et inconvénients.

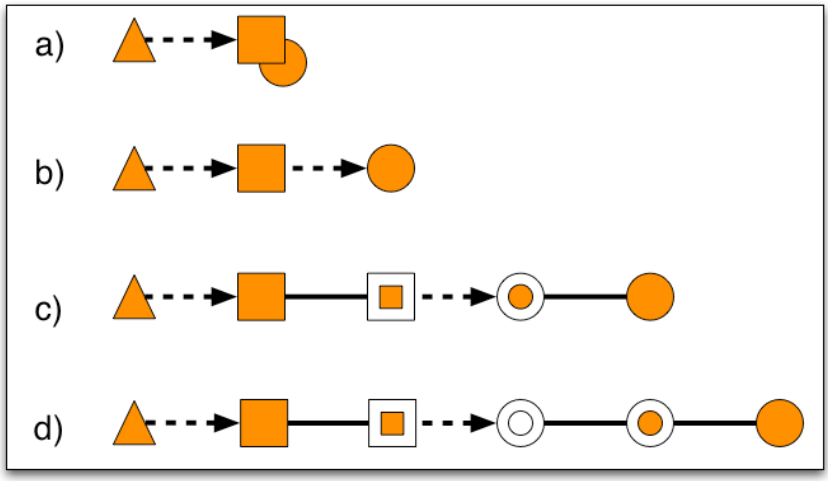

Figure 18. *Quatre grandes classes de techniques d'interaction*

a) L'utilisateur agit sur un instrument qui est également la ressource d'interaction à coupler. Les Smart-its Friends (Holmquist *et al.*, 2001) font clairement partie de cette catégorie. L'utilisateur tient dans ses mains les objets à coupler . Cette classe de technique présente l'avantage que la charge cognitive pour l'utilisateur quant à la désignation des ressources est quasi-nulle. Cependant, ce genre de technique n'est applicable que pour des dispositifs très mobiles.

b) L'utilisateur, à l'aide d'un instrument désigne les ressources qu'il désire coupler. Dans les DataTiles, l'utilisateur se sert d'un stylet pour désigner les deux tuiles qu'il désire coupler. Ici, l'action se situe toujours dans le monde physique, que se soit pour l'instrument ou la ressource à coupler. Cette physicalité, présente les même intérêts que la classe de technique précédente au niveau cognitif et rend plus souple son utilisation avec des dispositifs moins mobiles. Un niveau d'indirection est cependant ajouté.

c) L'utilisateur manipule un instrument numérique grâce à un instrument physique. Ensuite l'instrument numérique agit sur un autre objet numérique représentant la ressource à coupler. La technique Click&Couple (Barralon *et al.*, 2005) (décrite en 6.4.1) réalisé avec une souris en est l'exemple parfait. La souris, manipule le curseur qui agit sur le window manager de la surface à coupler. Dans cette classe de technique, l'utilisateur doit faire l'association entre l'instrument physique et sa représentation, la ressource d'interaction et sa représentation et le fait que l'interaction se passe entre les deux représentations. Si cette classe semble plus lourde cognitivement pour l'utilisateur, elle est néanmoins très flexible quant au type de ressources utilisées ; cette classe se présente comme une bonne alternative lorsque les contraintes (matérielles par exemple) empêchent l'utilisation des techniques a) et b).

d) Dans cette dernière classe, un niveau supplémentaire est ajouté par rapport aux techniques c) puisque la désignation de la ressource se fait par une représentation numérique de la représentation de la ressource. Les configurateurs, comme celui présenté en Figure 20 ou dans (Barralon *et al.*, 2005) incarnent cette classe de technique. C'est par le biais d'une représentation du window manager que la désignation est réalisée. Cette classe de techniques semble être la plus lourde cognitivement (lors de nos utilisations du système décrit), mais c'est également celle qui offre un contexte d'utilisation le plus vaste. On peut citer par exemple le couplage

à distance de ressources d'interaction ou la manipulation d'un grand nombre de ressources.

Les quatre classes de techniques ci-dessus ont toutes une implémentation dans des prototypes de recherche. L'utilisation de la notation permet de mettre en évidence deux autres classes de techniques. La Figure 19 représente ces deux classes que nous ne pouvons donc pas illustrer. Une étude plus approfondie des avantages et inconvénients de chaque groupe de décomposition (physique – représentation ou physique – représentation – représentation, etc.) pourrait conduire à la création de techniques d'interaction répondant à des besoins spécifiques.

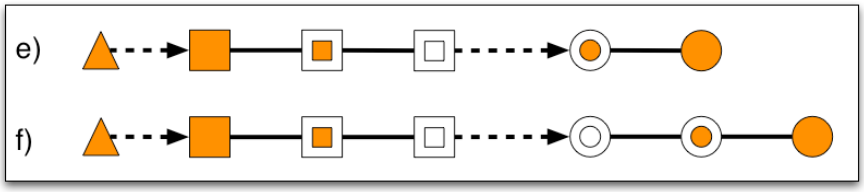

Figure 19. *Classes de techniques d'interaction non présentes dans les prototypes de recherche.*

En illustrant les classes de techniques, nous ne pouvons pas faire l'hypothèse qu'une technique d'interaction ne fait partie que d'une seule classe. Nous rappelons que le couplage nécessite la désignation de deux ressources d'interaction. On peut donc envisager qu'une des ressources sera désignée suivant une classe, et l'autre, suivant une autre classe créant quinze types de techniques d'interaction en considérant 6 classes ($C_6^2 = A_6^2/2! = 15$) et 6 types si on ne considère que les 4 classes illustrées.

Nous nous intéressons maintenant au deuxième rôle que remplit une technique d'interaction : la spécification de la fonction obtenue par le couplage.

5.2 Technique monofonctionnelle / plurifonctionnelle

Parmi les prototypes de recherches, de nouvelles techniques d'interactions sont apparues. Dans DataTiles, l'utilisateur agence des tuiles transparentes sur un écran à cristaux liquides. Le couplage de deux tuiles se fait en traçant à l'aide d'un stylet particulier, un trait entre deux tuiles adjacentes. Dans Synchronous Gesture, deux tablettes équipées d'accéléromètres sont choquées l'une contre l'autre pour échanger des informations. En choquant une seule tablette sur une autre « fixe », la première étend son affichage sur la seconde. Tandler avec les ConnectTables propose de coupler des surfaces en mettant en contact leur haut pour entamer une collaboration. Enfin, Rekimoto et ses collègues (2003a) présentent SyncTap comme technique pour le couplage de dispositifs mobiles.

Ce bref récapitulatif montre que deux catégories de techniques d'interaction sont identifiables. Certaines, dites mono-fonctionnelles, permettent la mise en œuvre d'une seule fonction de couplage. Par exemple, la connexion de deux ConnectTables permet systématiquement l'échange des contenus des deux surfaces. De même, dans DataTiles, le couplage (par un trait) de deux tuiles produit la connexion des données de sorties de l'une des tuiles avec les données d'entrées de l'autre tuile. Du point de vue utilisateur, ces techniques offrent l'avantage d'être faciles à utiliser et par définition prévisibles. On leur oppose les techniques plurifonctionnelles.

Cet autre type de techniques s'illustre par exemple avec Synchronous Gesture. Cette technique est dérivée pour obtenir trois fonctions différentes : l'annexion de surface, le partage réciproque d'information et le partage unilatéral. Une technique

permettant plusieurs fonctions semble plus intéressante qu'une monofonctionnelle car un seul apprentissage couvre plusieurs utilisation. Cependant, une vigilance s'impose car des erreurs sont alors possibles, par exemple entre choquer deux tablettes ensemble et n'en choquer qu'une avec l'autre. Dernier exemple de technique plurifonctionnelle, le configurateur de surfaces fourni avec les systèmes d'exploitation actuels (voire Figure 20). Cette technique autorise deux fonctions : par défaut l'extension de la surface principale sur la surface secondaire ou le choix de répliquer l'affichage sur la surface secondaire.

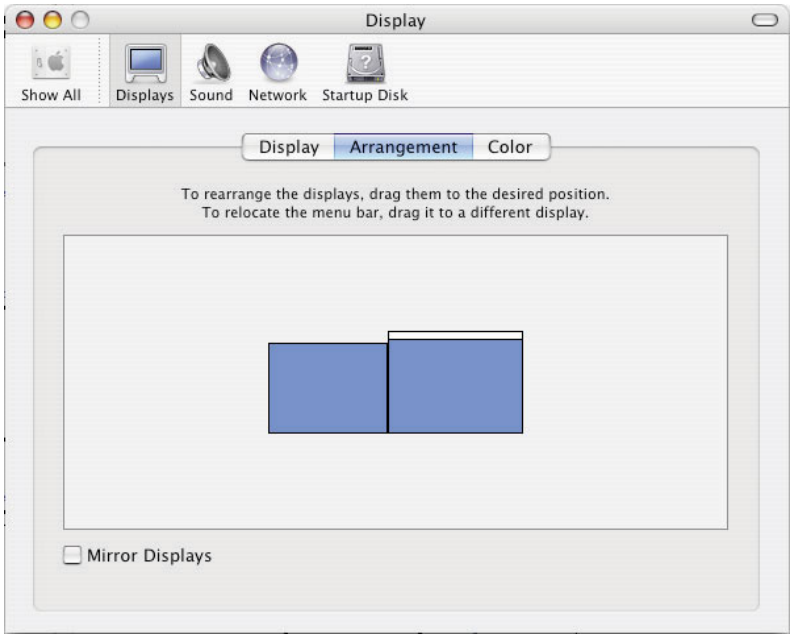

Figure 20. *Configurateur de surfaces sous Mac Os X pour les fonctions "extension de la surface d'affichage" ou "mirroring"*

En synthèse, certaines techniques sont dédiées à une fonction, faciles à apprendre et dont le risque d'erreur est moindre, d'autres, peut-être plus difficiles d'utilisation, présentent l'avantage de produire plusieurs fonctions. En faisant le lien avec les fonctions génériques que nous avons évoquées, nous traitons maintenant des techniques génériques présentant un lien certain avec les techniques plurifonctionnelles.

5.3 Techniques génériques

Par définition, une technique générique porte sur une fonction générique ou sur des ressources d'interaction génériques. Une ressource d'interaction générique est par exemple, un dispositif de pointage, et sera instancié en souris ou stylet.

Dans (Holmquist *et al.*, 2001) de petits dispositifs physiques, équipés d'accéléromètres, se connectent lorsqu'ils détectent qu'ils sont agités ensemble. Cette technique générique peut être énoncée ainsi : « Agiter deux ressources d'interaction pour les coupler ». Dans ce même article, deux dispositifs sont utilisés pour fabriquer une alarme qui se déclenche lorsqu'un enfant est trop éloigné de ses parents. Enfants et parents portent des dispositifs qui ont été couplés, indiquant que c'est leur distance relative qui déclenche l'alarme.

De façon similaire, si l'on reprend l'idée de Synchronous Gesture, on construit les techniques suivantes :

1. « Choquer deux ressources d'interaction pour les coupler »
2. « Choquer deux ressources d'interaction pour échanger leur fonction » (qui hérite de 1)
3. « Choquer deux surfaces pour échanger leur affichage » (qui hérite de 2)
4. « Choquer deux souris pour permuter l'affectation des pointeurs » (qui hérite également de 2)

Cette décomposition apporte un bénéfice certain du point de vue utilisateur, pour lequel l'apprentissage peut être réduit. Cependant, le premier niveau que nous proposons n'apporte que peu d'intérêt sinon de restreindre cette technique uniquement aux dispositifs mobiles. Pour le développeur, la capitalisation est également indéniable. Dans le cas de Synchronous Gesture, la mise en œuvre du module détectant les gestes synchronisés serait réutilisé, qu'il s'agisse de surfaces ou d'instruments.

Enfin, un lien existe entre les techniques plurifonctionnelles et les techniques génériques. Une technique plurifonctionnelle est l'union d'une technique générique et de plusieurs de ces techniques filles. Par exemple, un configurateur du type de (Barralon *et al.*, 2005) autorisant les fonctions d'extension d'affichage et de duplication est une technique unissant :

- « Coupler deux surfaces par leur représentation numérique »
- « Étendre l'affichage d'une surface sur une autre par la proximité de leur représentation numérique »
- « Dupliquer l'affichage d'une surface sur une autre en superposant leur représentation numérique »

6 Eléments de mise en œuvre

La Figure 21 représente les niveaux de service que nous avons développés pour répondre au problème du couplage de ressources d'interaction hétérogènes tout en permettant la migration des IHM au niveau du pixel. Dans la couche du bas, les capteurs physiques détectent la proximité d'entités physiques jouant le rôle de surface. Les contexteurs fournissent les services logiciels de découverte et de localisation de ces ressources. I-AM prend en charge les mécanismes de base pour le couplage et la migration et la distribution des IHM au niveau du pixel. Au niveau le plus haut, Ambient-Desktop s'appuie sur les services d'I-AM et des contexteurs pour assurer le couplage de bureaux. Dans les sections qui suivent, nous présentons ces services en détail.

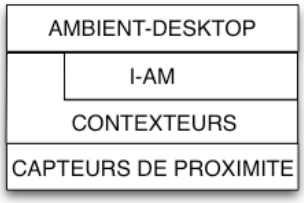

Figure 21. *Décomposition fonctionnelle de notre solution*

6.1 Les Capteurs de Proximité

Nous avons développé notre propre capteur de proximité pour les raisons suivantes :

- La solution RFID (Radio Frequency Identification) proposée par les ConnecTables est difficile à mettre en œuvre car elle nécessite l'utilisation d'antennes dont la consommation électrique serait trop pénalisante en situation mobile.
- Les accéléromètres utilisés par Hinckley, permettent uniquement de détecter les côtés mis en jeu par le couplage de surface.

Nous développons donc un capteur à faible consommation pour une utilisation en mobilité, mais pouvant fournir plus de précision concernant les relations spatiales entre les surfaces. Un module satellite est placé sur chaque bord d'écran (voir Figure 22). Il y a détection de proximité lorsque deux satellites appartenant à des écrans différents « se voient ».

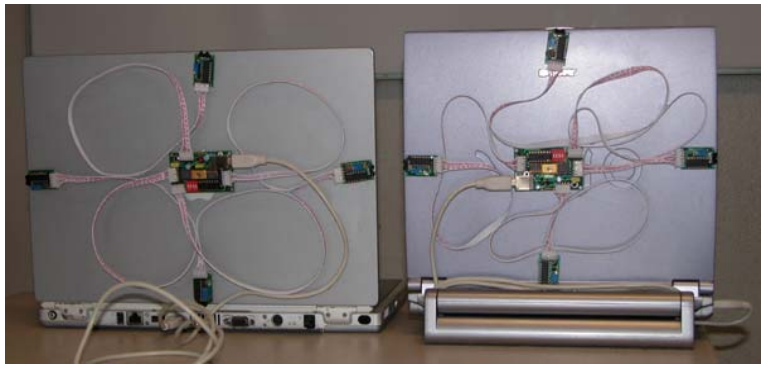

Figure 22. *Portables équipés de notre capteur de proximité*

Sur le plan logiciel, un ProximitySurfacesContextor, fonctionnant sur la machine hôte, pilote le capteur de proximité. Il informe le capteur de la machine sur laquelle il fonctionne, de son identifiant d'entité et réceptionne les données de couplage du capteur. Une description plus détaillée du fonctionnement de ces capteurs est faite dans Barralon *et al.* (2005). Au final, ces capteurs assurent la découverte physique d'autres capteurs en s'identifiant mutuellement. Ces informations sont ensuite transmises aux contexteurs.

6.2 Les Contexteurs

Les contexteurs forment une infrastructure répartie de capture du contexte d'interaction (Rey, 2005). Cette infrastructure s'auto construit dynamiquement par assemblage d'agents logiciels, les contexteurs, qui, ensemble, fournissent les données contextuelles requises par des processus clients. Dans le cadre du travail présenté ici, I-AM est un tel client. Ce client est intéressé de connaître, à tout instant, l'ensemble des surfaces disponibles dans l'espace d'interaction, de même que les relations spatiales entre ces surfaces.

Sur chaque machine IP couplable, un SurfacesContextor s'exécute en permanence pour scruter l'arrivée et le départ des entités physiques reliées à cette machine jouant le rôle de surface. Par défaut, tout écran joue le rôle de surface. De plus, pour chacun des écrans d'une machine, le SurfacesContextor fournit sa résolution actuelle en pixels, sa taille en cm, la taille de chacun de ses bords ainsi que

le type (TFT/CRT) de l'écran. L'utilisation de ces données par I-AM est décrite dans Lachenal (2004).

Pour l'application qui nous intéresse, l'infrastructure des contexteurs doit également détecter les relations spatiales entre les surfaces et notamment la proximité de deux surfaces. L'idée, reprise du scénario de l'introduction, est de coupler des surfaces par proximité afin d'obtenir un espace d'affichage unifié. Pour cela, l'infrastructure inclut pour chaque machine IP :

- un ProximitySurfacesContextor qui encapsule le capteur de proximité décrit ci-dessus, ou
- un IAMLinksContextor basé sur un capteur logiciel.

Ces deux contexteurs envoient à leurs clients (dans notre cas I-AM) le même type d'information, c'est-à-dire la liste des liens entre la machine où s'exécutent le contexteur et ses voisins.

6.3 I-AM (Interaction Abstract Machine)

I-AM est une machine du même niveau d'abstraction que les gestionnaires graphiques de base comme X window. Ces gestionnaires gèrent les ressources d'interaction d'une station de travail dont ils assurent le partage entre les applications clientes. Comme ces gestionnaires, I-AM est un intergiciel orienté Interaction Homme-Machine, mais I-AM en étend la couverture fonctionnelle pour tenir compte des requis de l'informatique ambiante. En effet, I-AM permet la construction dynamique d'espaces interactifs dont les surfaces et les instruments sont gérés par des stations de travail distinctes exécutant des systèmes d'exploitation éventuellement différents (MacOS, Windows), et donne l'illusion, au programmeur comme à l'utilisateur, que ces ressources sont gérées de manière uniforme par une seule station de travail. Plus spécifiquement, I-AM :

1. Masque l'hétérogénéité du matériel et des systèmes d'exploitation sous-jacents au bon niveau d'abstraction. Cette fonction est assurée en projetant les services d'I-AM dans les termes des systèmes d'exploitation sous-jacents.
2. Intègre les services de découverte dynamique de ressources d'interaction grâce à l'infrastructure de contexteurs.
3. Modélise les relations spatiales entre les ressources d'interaction (et en déduit une topologie des surfaces). Ce service est assuré grâce aux informations fournies par l'infrastructure de contexteurs.
4. Permet la distribution et la migration des IHM graphiques au niveau du pixel. Ce service est assuré par une fonction de projection (une transformation affine) entre les espaces numériques et la topologie des surfaces.

Une description détaillée d'I-AM est présentée dans Lachenal (2004). Ici, nous en montrons l'exploitation avec la mise en œuvre d'Ambient-Desktop.

6.4 Ambient-Desktop

Ambient-Desktop offre à l'utilisateur des techniques d'interaction pour configurer à façon son espace de travail. La (re)configuration s'effectue par couplage de surfaces, avec pour conséquence, le couplage de bureaux (au sens desktop des stations de travail usuelles).

Ambient-Desktop se doit de garantir l'observabilité du couplage (l'utilisateur doit pouvoir en évaluer l'état), la curabilité (un couplage non désiré doit pouvoir être défait), la prévisibilité (l'utilisateur doit pouvoir prédire l'effet d'un couplage) et l'honnêteté (l'état observable d'un couplage doit être conforme à l'état interne et doit être compris correctement par l'utilisateur) comme nous l'avons exprimé en section 3.5.

Dans les sections qui suivent, nous présentons les techniques d'interaction pour le couplage/découplage de surfaces, conçues dans le respect de ces propriétés d'utilisabilité. Nous détaillons notamment ces techniques grâce à la notation graphique introduite en 5.1. avant de rappeler brièvement comment se déroule la génération des couplages dans notre implémentation.

6.4.1 Click&Couple

Cette technique, décrite dans (Barralon *et al.*, 2004) est basée sur la détection d'événements souris synchronisés. En utilisant simultanément une souris sur chacune des deux surfaces à coupler, l'utilisateur réalise deux « click » synchronisés. L'effet immédiat est l'ouverture d'une passerelle entre les deux espaces interactifs. Le halo délimitant les frontières de l'espace interactif présente maintenant un passage entre les deux surfaces. De plus, un feedback apparaît sur les deux surfaces (Figure 23) à la manière des tablettes de Hinckley (2003) pour symboliser la liaison. En agissant sur ce feedback, il est possible de casser les liaisons, donc de découpler. Le retour d'information que nous venons d'énoncer est commun aux trois techniques d'interaction que nous décrivons.

Figure 23. *Feedback de couplage en deux écrans*

La description graphique de cette technique est donnée en Figure 24. L'utilisateur agit sur une souris, qui par le biais de son curseur agit sur le window manager de l'écran à coupler. Comme nous l'indiquons en section 5.1.2 Click&Couple est une technique dont les désignations des deux ressources d'interaction font partie de la même classe. Cette technique se révèle particulièrement utile lorsqu'il n'est pas possible de modifier la proximité physique des deux écrans que nous présentons ci-dessous ou en situation multi-utilisateurs : problème de synchronisation des deux utilisateurs.

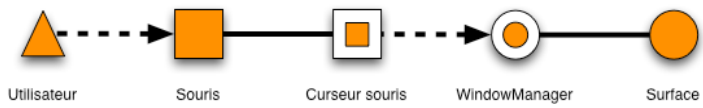

Figure 24. *Description de la technique Click&Couple*

6.4.2 Couplage par proximité de surfaces

La deuxième technique possible est l'utilisation des détecteurs de proximité précédemment décrits (Figure 22). Pour coupler, l'utilisateur doit rapprocher physiquement les deux écrans et aligner deux modules satellites. Le couplage des surfaces équipées est immédiatement réalisé. La position physique des capteurs ainsi que leur champ d'action n'autorise que la création de mosaïques de surfaces comme les DataTiles (Rekimoto *et al.*, 2001) ; deux surfaces ne peuvent se superposer. Le découplage s'effectue simplement par l'éloignement des deux surfaces. On se place ici clairement dans l'interaction proximale (Rekimoto *et al.*, 2003b) dont peuvent se passer le Click&Couple et le configurateur. En effet, ces deux techniques permettent de simuler la proximité sans pour autant que les surfaces soient physiquement en contact. L'utilisateur manipule donc un intrument qui se révèle être la surface à coupler (Figure 25). Cette technique est particulièrement adaptée lors de l'utilisation de dispositifs mobiles.

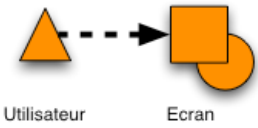

Figure 25. *Description de la technique avec capteurs de proximité*

6.4.3 Configurateur

Cette dernière technique utilise une IHM semblable, sur certains points, à ceux que l'on trouve pour l'ajout d'un écran sur la plupart des systèmes d'exploitation.

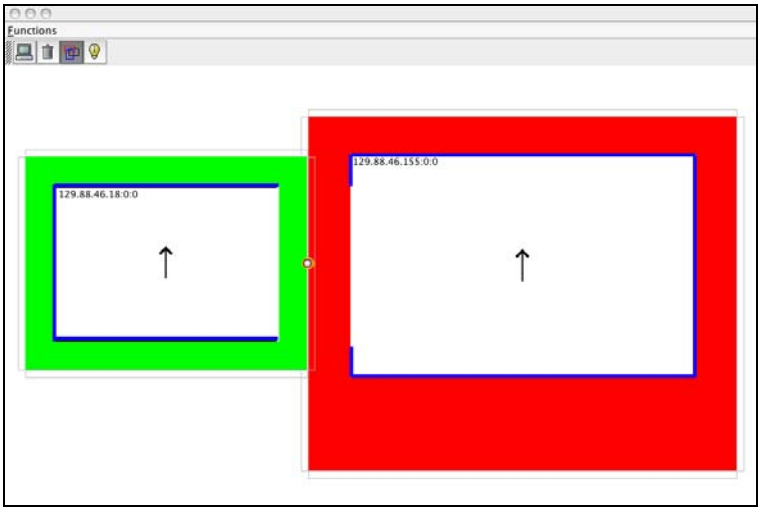

Figure 26. *Configurateur avec deux surfaces, leurs bords physiques et une liaison*

Le configurateur exploite les contexteurs pour détecter l'arrivée et le départ de surfaces. Tout comme les configurateurs actuels, la manipulation directe des écrans permet de gérer leur arrangement spatial. La Figure 26 montre deux surfaces couplées. Le configurateur permet la rotation des écrans, la visualisation des bords physiques et des frontières des surfaces, la création automatique des liaisons associées à l'arrangement en cours et la création automatique de liaisons prédéfinies.

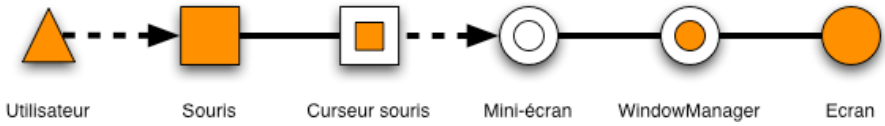

Figure 27. *Description du configurateur*

La Figure 27 montre comment fonctionne le configurateur au regard de la désignation des ressources d'interaction. L'utilisateur agit sur la souris dont le curseur manipule des représentations numériques (mini-écrans) des windows managers (représentation numérique d'un objet physique) des écrans à coupler. Cette technique peut être utilisée en toutes circonstances. L'effort cognitif, lié par exemple à l'association des mini-écrans et des écrans physiques, étant important on lui préférera l'une des deux autres techniques.

6.4.4 Génération des couplages

Les trois techniques que nous venons d'exposer reposent sur I-AM. Aussi, dans un premier, les couplages déduits des trois techniques d'interaction sont identiques.

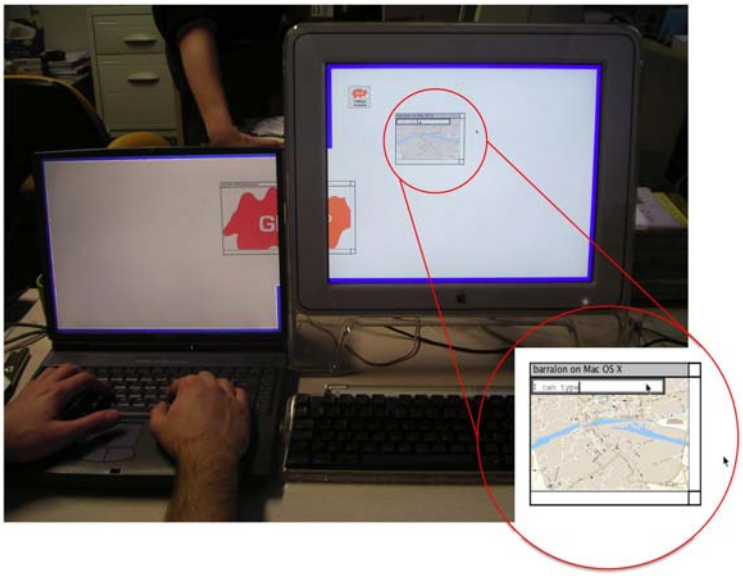

Figure 28. *Un espace d'interaction construit avec I-AM à partir des ressources d'interaction d'un PC portable exécutant Windows XP et d'un MacIntosh exécutant MacOSX.*

La Figure 28 que nous avons déjà présentée offre une représentation des couplages obtenus. L'espace d'affichage est considéré comme ne faisant qu'un. Aussi la souris et le clavier du portable de gauche peuvent indifféremment agir sur les deux écrans tout comme la souris et le clavier de la station de droite.

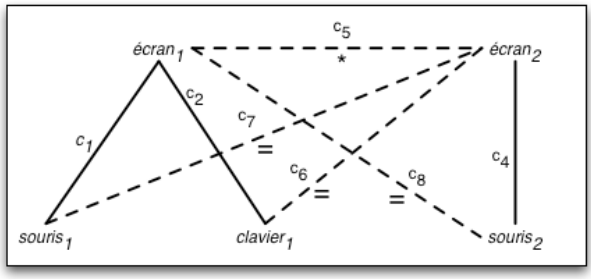

Figure 29. *Configuration terminale avec I-AM.*

La génération des couplages est semblable à celle que nous avons proposée en 4.1 (cas 3, Figure 11) et nous la résumerons à l'aide du formalisme mathématique :
Configuration initiale : [$c_1 + c_2 + c_3 + c_4 + c_5 + c_6$]
Couplage initiateur : c_7

Règle d'ajout :
$c_7 + [c_1 + c_2 + c_3 + c_4 + c_5 + c_6] = [c_7 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6]$

Règle de génération :
$c_7 * [c_7 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6]$
$= c_7 * c_7 + c_7 * c_1 + c_7 * c_2 + c_7 * c_3 + c_7 * c_4 + c_7 * c_5 + c_7 * c_6$
$= \varnothing + c_8 + c_9 + \varnothing + c_{10} + c_{11} + \varnothing$

Règle d'ajout :
$c_8 + c_9 + c_{10} + c_{11} [c_7 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6]$
$= [c_8 + c_9 + c_{10} + c_{11} + c_7 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6]$

Règle de génération :
$(c_8 + c_9 + c_{10} + c_{11}) * [c_8 + c_9 + c_{10} + c_{11} + c_7 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6]$
Pour plus de lisibilité, on scinde le calcul en quatre

1. $c_8*c_8+c_8*c_9+c_8*c_{10}+c_8*c_{11}+c_8*c_7+c_8*c_1+c_8*c_2+c_8*c_3+c_8*c_4+c_8*c_5+c_8*c_6$
2. $+ c_9*c_8+c_9*c_9+c_9*c_{10}+c_9*c_{11}+c_9*c_7+c_9*c_1+c_9*c_2+c_9*c_3+c_9*c_4+c_9*c_5+c_9*c_6$
3. $+ c_{10}*c_8+c_{10}*c_9+c_{10}*c_{10}+c_{10}*c_{11}+c_{10}*c_7+c_{10}*c_1+c_{10}*c_2+c_{10}*c_3+c_{10}*c_4+c_{10}*c_5+c_{10}*c_6$
4. $+ c_{11}*c_8+c_{11}*c_9+c_{11}*c_{10}+c_{11}*c_{11}+c_{11}*c_7+c_{11}*c_1+c_{11}*c_2+c_{11}*c_3+c_{11}*c_4+c_{11}*c_5+c_{11}*c_6$

1. $\varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$
2. $+ \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$
3. $+ \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$
4. $+ \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing + \varnothing$

Configuration terminale : [$c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8 + c_9 + c_{10} + c_{11}$]

7 Conclusion et perspectives

En informatique conventionnelle, le couplage de ressources d'interaction est maîtrisé car les ressources d'interaction sont en nombre limité et que les couplages sont prédéfinis. En informatique ambiante, la mobilité des utilisateurs et des ressources d'interaction ainsi que l'intégration à l'environnement physique de ces

dernières posent de nouveaux challenges pour le concepteur d'applications mais aussi pour l'utilisateur final.

Dans cet article, nous proposons donc des bases pour une étude du couplage de ressources d'interaction. Dans un premier temps, le cycle de vie du couplage, construit sur la base des propriétés ergonomiques permet l'analyse mais aussi la conception de systèmes prenant en charge le couplage dynamique de ressources d'interaction. Dans un deuxième temps, nous explorons les impacts de l'ajout d'un couplage en fournissant un formalisme graphique et un formalisme mathématique pour raisonner sur la génération de couplage. Dans cette optique, des heuristiques de générations sont à trouver et les fonctions génériques semblent être une piste prometteuse.

Notre analyse se poursuit par l'étude des techniques d'interaction sous ses deux aspects : l'aspect fonctionnel avec les techniques mono et plurifonctionnelles et sous l'aspect de la désignation des ressources d'interaction. Pour ce dernier point, l'utilisation d'une notation graphique fait apparaître quatre classes de techniques d'interaction dans l'état de l'art, mais également de deux classes non représentées.

Enfin nous proposons une mise en œuvre logicielle qui couvre les deux axes de cet article : le couplage dynamique et la génération de couplages, ainsi que les techniques d'interaction.

En perspectives à ce travail, l'étude des fonctions génériques et leurs déclinaisons pour des types de ressources d'interaction usuelles permettra à court terme de vérifier si elles permettent la génération automatique de couplages. D'autres heuristiques viendront certainement s'ajouter pour espérer obtenir un processus de génération automatique. Une analyse plus fine des techniques d'interaction peut également apporter des éléments. Dans notre implémentation par exemple, le Click&Couple se destine à un usage mono-utilisateur alors que les capteurs de proximité peuvent être utilisés par plusieurs utilisateurs. Dans le premier cas, une heuristique pourrait être « la fusion des souris et des claviers » ; puisque qu'un seul utilisateur est présent, qu'il n'a qu'un seul espace d'interaction, il n'aurait alors besoin que d'un seul curseur souris et une seule file d'événements clavier.

Au final, il est également primordial d'étudier le découplage suivant les mêmes axes que l'étude que nous venons de présenter. D'une part les impacts de la suppression d'un couplage en fournissant des mécanismes proches du formalisme graphique et mathématique, et d'autres part les techniques d'interaction pour le découplage.

8 Remerciements

Nous tenons tout particulièrement à remercier Joëlle Coutaz pour son aide à la rédaction de cet article ainsi que Gaëtan Rey, Christophe Lachenal et Viet-Tung NGuyen pour leur collaboration dans la mise en place du prototype Ambient-Desktop.

9 Bibliographie

Barralon, N., Lachenal, C., Coutaz, J. (2004). Couplage de ressources d'interaction. In *Proceedings of IHM'04, International Conference Proceedings Series*, Namur, Belgium, 13-20, ACM Press.

Barralon, N., Nguyen, VT., Rey, G. (2005). Techniques de couplage de bureaux : Ambient-Desktop comme illustration. In *Proceedings UbiMob'05*, 193-200, Grenoble, 31 mai-3 juin.

Bastien, J.M.C., Scapin, D. (1993). *Ergonomic Criteria for the Evaluation of Human-Computer interfaces*. Rapport n°156, INRIA, France.

Bérard, F. (2003). The Magic Table: Computer-Vision Based Augmentation of a Whiteboard for Creative Meetings. In *Proceedings IEEE workshop on Projector-Camera Systems*, Nice, France, october.

Borkowski, S., Riff, O., Crowley, J.L. (2003). Projecting rectified images in an augmented environment. In *Proceedings ProCams Workshop, International Conference on Computer Vision, ICCV 2003*, Nice, France, october.

Bowman, D.A., Hodges, LF. (1999). Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. *Journal of Visual Languages and Computing*, 10, 37-53.

Bellotti, V., Back, M., Edwards, K., Grinter, R., Henderson, A., Lopes, C. (2002). Making Sense of Sensing Systems: Five Questions for Designers and Researchers. In *Proceedings of CHI 2002*, 415-422, ACM Press, New York.

Coutaz, J., Nigay, L., Salber, D. (1995). Multimodality from the user and system perspectives, In *Proc. ERCIM (European Research Consortium for Informatics and Mathematics), workshop on User Interface For All*, Heraklion, September.

Dragicevic, P. (2004). *Un modèle de configurations d'entrée pour des systèmes interactifs multi-dispositifs hautement configurables*. Thèse de l'Université de Nantes, mars.

Foley, JD., van Dam, A., Feiner, SK., Hughes, JF. (1990). *Computer Graphics Principles and Practice*. Second Edition, Addison-Wesley Publishing Company.

Gram, Ch., Cockton, G. (1996). *Design Principles for Interactive Software*. Chapman & Hall, London.

Hinckley, K., (2003). Synchronous gestures for multiple persons and computers. In *Proceedings UIST'03*, ACM Press, 149-158.

Holmquist, L.E., Mattern, F., Schiele, B., Alahuhta, P., Beigl M., Gellersen, H.W (2001). Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefact. In *Proceedings Ubicomp 2001*, Atlanta, Georgia, 116-122.

Izadi, S., Brignull, H., Rodden, T., Rogers, Y., Underwood, M. (2003). Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media. In *Proceedings UIST'03*, ACM Press, 159-168.

Lachenal, C. (2004). *Modèle et Infrastructure Logicielle pour l'Interaction multi-instrument, multi-surface*. Thèse de l'Université Joseph Fourier, décembre.

Lecolinet, E. (2003). A molecular architecture for creating advanced GUI. In *Proceedings UIST'03*, ACM Press, 135-144.

Myers, B., Stiel, H., Gargiulo, R. (1998). Collaboration Using Multiple PDAs Connected to a PC. In proceedings of CSCW98, ACM Press, 285-294.

Nabaztag. *Violet, the smart Object Company*, Disponible à : http://www.violet.net/ et http://www.nabaztag.com/.

Norman, D. A., Draper, S.W. (1986). *User Centered System Design*. Lawrence Erlbaum Associates, Hillsdale.

Rekimoto, J., Ullmer, B., Oba, H. (2001). DataTiles: A Modular Platform for Mixed Physical and Graphical Interactions. In *Proceedings of ACM CHI'01 Conference on Human Factors in Computing Systems*, ACM Press, 269-276.

Rekimoto, J., Ayatsuka, Y., Kohno, M. (2003a). SyncTap: An Interaction Technique for Mobile Networking. In *Proceedings of MOBILE HCI 2003*, Udine, Italy, 104-115.

Rekimoto, J., Ayatsuka, Y., Kohno, M., Oba, H. (2003b). Proximal Interactions: A Direct Manipulation Technique for Wireless Networking. In *Proceedings of Interact'03*, Zürich, 511-518.

Renevier, P., (2004). *Systèmes Mixtes Collaboratifs sur Supports Mobiles : Conception et Réalisation*. Thèse de l'Université Joseph Fourier, Juin.

Rey, G., (2005). *Contexte en Interaction Homme-Machine : le contexteur*. Thèse de l'Université Joseph Fourier, Août.

Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N., Steinmetz, R. (2001). ConnecTables: Dynamic Coupling of Displays for the Flexible Creation of Shared Workspaces. In proceedings of UIST'01, ACM Press, 11-20.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, september, 94–104.