Ingénierie des Systèmes Interactifs

1 - Introduction

Michel Beaudouin-Lafon, mbl@lri.fr

Introduction

Ce cours présente les principes, méthodes et outils pour le développement de systèmes interactifs. L'accent est mis sur la notion de conception centrée sur l'utilisateur, qui implique un style itératif de conception-développement-évaluation, et sur les applications utilisant des interfaces graphiques.

Définitions

Un *système interactif* est une application informatique qui prend en compte, au cours de son exécution, d'informations communiquées par le ou les utilisateurs du système, et qui produit, au cours de son exécution, une représentation perceptible de son état interne. Typiquement, les entrées fournies par l'utilisateur dépendent des sorties produites par le système et inversement.

Les systèmes interactifs se distinguent donc des applications de type "batch" qui lisent, au début de leur exécution, des données prédéfinies, et produisent à la fin de leur exécution des résultats. Du point de vue informatique, un système interactif est donc un système ouvert : les entrées fournies au système dépendent de ses sorties, d'une façon qui n'est pas accessible au système.

De nos jours, les systèmes interactifs constituent une part croissante des applications informatiques. Comme nous le verrons dans ce cours, la conception, le développement et l'évaluation des systèmes interactifs nécessitent des méthodes et des techniques particulières afin de prendre en compte l'utilisateur.

Classiquement, on décompose un système interactif en deux parties : *l'interface utilisateur* et le *noyau fonctionnel*. L'interface utilisateur contient les éléments logiciels et matériels dédiés à la capture des entrées de l'utilisateur et à la restitution des sorties du système. Le noyau fonctionnel contient le reste du système, c'est-à-dire ses composants de calcul et de stockage de l'information.

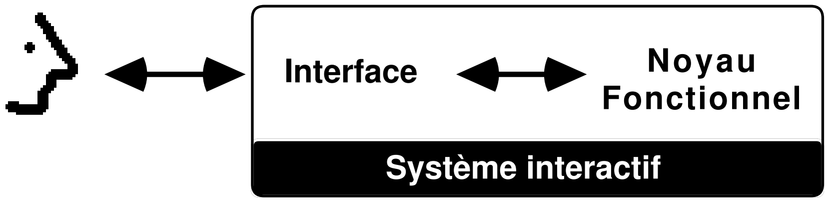


Figure 1 : décomposition d'un système interactif

Cette décomposition de doit pas laisser croire que ces deux parties sont forcément des modules logiciels indépendants, ni surtout qu'elles peuvent être conçues indépendamment. S'il est vrai que l'on est souvent conduit à développer des interfaces pour des noyaux fonctionnels existants ("legacy systems"), la seule façon de développer des systèmes interactifs de qualité est de concevoir l'ensemble du système, en prenant en compte les caractéristiques de l'utilisateur au même titre que les contraintes techniques du noyau fonctionnel.

Exemple : un shell iconique pour Unix

Supposons que l'on veuille réaliser un système interactif (shell iconique) permettant à un utilisateur du système Unix de manipuler le système fichiers grâce à des icones, comme avec le Finder du Macintosh. Le noyau fonctionnel, ici pré-existant, est le système de gestion de fichiers d'Unix. Deux problèmes principaux se posent : la gestion de la cohérence et la gestion des types de fichiers.

L'interface iconique doit être capable de présenter une vue à jour du système de fichiers : à tout instant, les icones visibles dans une fenêtre doivent correspondre aux fichiers présents dans un répertoire ("directory"). Or le système Unix (dans sa version "standard") ne peut signaler à un processus le changement de contenu d'un catalogue : si un utilisateur crée, détruit ou change le nom d'un fichier, le seul moyen pour un autre utilisateur de s'en rendre compte est de comparer le contenu du répertoire avant et après l'opération. Ainsi, le seul moyen pour le shell iconique de maintenir à jour ses fenêtres sera de demander périodiquement au noyau fonctionnel le contenu des répertoires correspondants. Cela a pour conséquence le risque d'incohérence entre les mises à jour. Ainsi, il sera possible à un utilisateur d'attraper un fichier qui vient juste d'être détruit. De plus, l'interrogation répétitive du noyau fonctionnel dégradera les performances à cause notamment des nombreux appels système engendrés.

La solution à ce premier problème serait d'ajouter au noyau fonctionnel la possibilité de notifier les changements d'état du système de fichiers. Cette fonctionnalité a d'ailleurs été ajoutée à certaines versions d'Unix pour les besoins des interfaces graphiques ...

Le second problème est qu'il n'y a pas de notion de type de fichier sous Unix (cela est d'ailleurs souvent vu comme l'un des points forts de ce système). Une interface iconique a besoin d'associer un type à chaque fichier pour présenter à l'utilisateur les commandes disponibles sur ce fichier. Par exemple un fichier source C doit pouvoir être édité ou compilé, tandis qu'un fichier exécutable doit pouvoir être lancé avec des arguments éventuels. La convention usuelle, sous Unix, est que le type de fichier est encodé dans son nom : foo.c contient normalement un source C. Cette convention est néanmoins insuffisante dans le cas général et un shell iconique devra stocker l'association entre nom de fichier et type dans une base de données annexe, ce qui complique la gestion.

Ici aussi, la solution à ce problème serait de permettre à des applications d'ajouter aux fichiers des informations annexes comme le type.

Modèle conceptuel d'un système interactif

Un système interactif, comme toute application informatique, contient un certain nombre de fonctionnalités, qui sont distribuées dans différents modules selon une architecture logicielle. La séparation interface/noyau fonctionnel évoquée plus haut résulte du partage des fonctionnalités d'un système interactif entre fonctions de l'interface et fonctions (internes) du système. Ce découpage fonctionnel peut être aisément raffiné : l'utilisateur doit pouvoir fournir des entrées aux systèmes (appelées commandes) et le système doit pouvoir présenter le résultat des commandes à l'utilisateur. Cela conduit au modèle conceptuel "générique" (c'est-à-dire indépendant de toute application) de la figure 2. Lors de la conception d'un système interactif particulier, ce modèle générique est détaillé en définissant plus précisément chacun de ses composants.

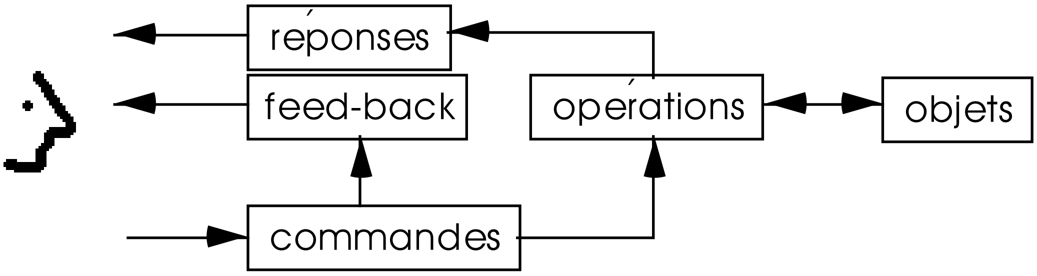


Figure 2 : modèle conceptuel générique

Dans ce modèle, les *objets* sont les objets du domaine de l'application : les mots, paragraphes et sections d'un éditeur de texte, les enregistrements et les relations d'une base de données, etc. Ces objets sont accédés et modifiés par des *opérations* internes, activées par l'utilisateur par l'intermédiaire de *commandes*. Par exemple, la commande de fermeture d'une fenêtre peut déclencher plusieurs opérations : sauvegarde du contenu de la fenêtre, déverrouillage d'enregistrements dans une base de données, etc.

Certaines commandes correspondent à une action instantanée de l'utilisateur : sélectionner un objet par pointage ou utiliser une touche de fonction par exemple. D'autres commandes nécessitent plusieurs actions élémentaires : faire un cliquer-tirer d'un icone ("drag-and-drop"), activer une commande dans un menu qui ouvre une boîte de dialogue, taper un commande textuelle suivie de RETURN, etc. Dans ce cas le système peut (doit) produire des sorties informant l'utilisateur qu'il fait bien ce qu'il pense faire : pendant le cliquer-tirer, l'icone du fichier suit le curseur de la souris, au fur et à mesure que l'on tape une commande, les caractères s'affichent, etc. Ce retour d'information est appelé *feed-back.*

Enfin, lorsque les opérations résultant de l'activation d'une commande sont exécutées, le système produit des *réponses* perceptibles par l'utilisateur : le cliquer-tirer ouvre une fenêtre, ou fait changer la forme de l'icone de la poubelle par exemple.

Conception centrée sur l'utilisateur

L'exemple du shell iconique pour Unix montre que les fonctionnalités du noyau fonctionnel dépendent du type d'interaction que l'on veut fournir à l'utilisateur. Le choix d'une interface iconique par rapport à l'interface conversationnelle du "shell" classique d'Unix est a priori motivé par un certain bénéfice attendu : les icones sont vues comme plus simples et intuitives à manipuler qu'un langage de commandes, spécialement pour des utilisateurs novices ou occasionnels.

Des connaissances, réelles ou supposées, des utilisateurs et de l'utilisation du système sont donc fondamentales pour concevoir l'interface qui sera proposée à l'utilisateur. Une fois le système réalisé, il faudra confronter les hypothèse réalisées lors de la conception avec l'utilisation effective du système. La prise en compte de l'utilisateur est donc nécessaire aussi bien dans les phases de conception que d'évaluation. On se rend souvent compte, lors de l'évaluation, que les hypothèses faites étaient erronées, ou que le style d'interaction choisi ne tient pas ses promesses, ou que les utilisateurs inventent des usages non anticipés du système.

Ceci conduit à intégrer la prise en compte des facteurs humains dans les modèles de développement des systèmes interactifs. On regroupe l'ensemble de ces modèles et des techniques associées sous le terme de *conception centrée sur l'utilisateur* ("user-centered design"). La figure 3 illustre le cycle de vie d'un logiciel interactif en mettant l'accent sur la prise en compte des utilisateurs.

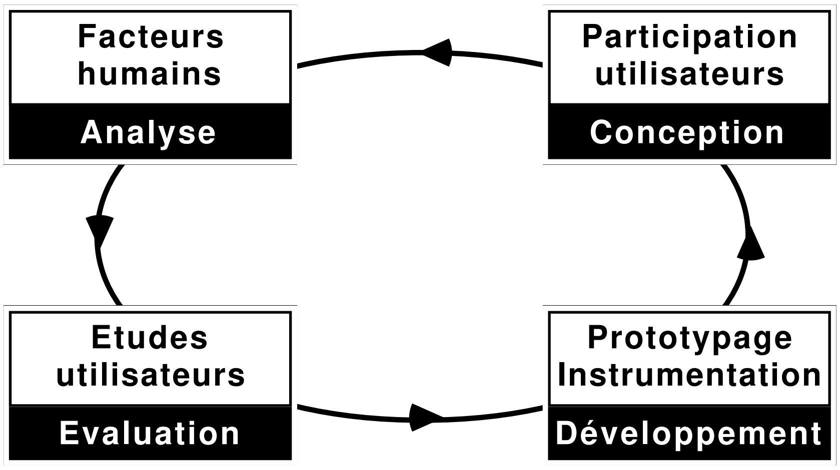


Figure 3 : conception centrée sur l'utilisateur

Lors de l'analyse, il s'agit notamment de prendre en compte les connaissances issues de la psychologie, d'analyser les outils existants, d'évaluer l'intérêt d'une solution informatique aussi bien au niveau des utilisateurs que de l'organisation, etc. Au niveau de la conception, il s'agit de définir un modèle conceptuel qui sera aussi facile que possible à appréhender par les utilisateurs, et si possible de faire participer les utilisateurs à la conception. Pendant le développement, il s'agit de pouvoir produire des prototypes permettant d'évaluer rapidement certains choix de conception. Enfin lors de l'évaluation il s'agit de vérifier, par des études avec des utilisateurs, si les hypothèse faites lors de la conception sont vérifiées dans le système final.

Si ce n'est pas le cas, il faut un refaire un cycle d'analyse-conception-développement-évaluation pour corriger les problèmes rencontrés. Ce cycle de *conception itérative* est inhérent aux systèmes interactifs : l'utilisateur humain et les modes d'utilisation ne sont pas formalisables d'une façon telle que l'on puisse les considérer comme fixes et connus. Quand bien même cela serait possible, l'usage d'un nouveau système par les utilisateurs change leurs modes de travail et donc leurs habitudes, leurs besoins et leurs attentes. Ainsi, on a *co-adaptation* entre le système et les utilisateurs : le système est conçu en fonction des besoins des utilisateurs, et ces besoins changent en fonction des caractéristiques du système. La seule façon d'atteindre un point d'équilibre est d'itérer la conception.

Bien entendu, l'utilisateur n'est pas le seul facteur à prendre en compte lors de la conception. Comme pour tout système informatique, il faut prendre en compte les facteurs techniques et organisationnels. Cependant, l'importance de la prise en compte des facteurs humains et le fait qu'ils imposent une cycle de développement itératif sont à l'origine du terme de "conception centrée sur l'utilisateur". Pour mettre en œuvre un tel style de conception, il faut un minimum de connaissances sur les utilisateurs, et pour commencer sur les capacités d'interaction des êtres humains. Ces connaissances nous sont fournies par la psychologie expérimentale. D'autres sources d'information sont utiles, comme les travaux issus de l'ergonomie des logiciels, sur lesquels nous auront l'occasion de revenir.

Eléments de psychologie appliqués aux systèmes interactifs

Cette section ne prétend pas être un cours de psychologie. Tout au plus s'agit-il de donner quelques éléments issus de la psychologie, pertinents pour la conception de systèmes interactifs.

L'être humain perçoit son environnement par l'intermédiaire des cinq sens. Il peut agir sur son environnement par l'intermédiaire de son système moteur (action physique) et par l'intermédiaire du langage. Enfin il est doté d'un système cognitif qui lui permet de mémoriser des informations et de planifier et contrôler des actions en fonction de ses perceptions et de sa mémoire.

Perception

Chez l'être humain, le sens principal est la vue. On considère que le traitement de données visuelles peut mobiliser jusqu'à la moitié du cerveau. L'ouïe et le toucher sont également bien développés, tandis que le goût et l'odorat sont moins développés (notamment si l'on compare à d'autres animaux). Il n'est donc pas surprenant de constater que la présentation d'information dans les systèmes interactifs est essentiellement visuelle. L'usage du son et du retour tactile commencent à être pris en compte, mais de façon très primitive et marginale. On peut estimer que la surcharge actuelle du canal visuel et la stagnation des technologies de présentation vont conduire à une exploitation croissante des autres sens.

Vue

Le système visuel a été très étudié par la psychologie expérimentale. Sur le plan physiologique, c'est un système très complexe et très spécialisé. Il est cependant loin d'être parfait : beaucoup d'animaux ont des systèmes visuels plus performants. Par ailleurs on peut "tromper" notre système visuel comme le démontrent de nombreuses illusions d'optique.

Le *champ* de la perception visuelle est proche de 180º. Cependant, les caractéristiques de la vision sont différentes entre le centre (focus d'attention) et la périphérie. Par exemple, la perception périphérique est moins sensible aux couleurs et plus sensibles aux mouvements (probablement parce que, dans les temps préhistoriques, il était important pour la survie de l'espèce de percevoir l'attaque d'un prédateur aussi tôt que possible dans le champ périphérique). Ainsi, il est souvent très difficile de repérer un oiseau dans un arbre, mais on le perçoit immédiatement au moment où il s'envole.

L'*acuité* visuelle mesure la capacité de résolution spatiale du système visuel, c'est-à-dire la taille minimale des objets discernables. Bien que cette taille dépende de nombreuses conditions, un trait noir de 0.04 mm d'épaisseur sur un fond blanc peut être distingué sans difficulté par un individu à une distance de 50 cm. Sur les écrans actuels, un trait d'un pixel a une épaisseur d'environ 0.3 mm, soit presque 10 fois l'acuité visuelle. Une imprimante à laser de qualité moyenne (300 points par pouce) permet de tracer des traits d'une épaisseur de 0.08 mm soit deux fois l'acuité visuelle.

La perception de la *couleur* est très complexe et dépendante du contexte. Ainsi, une même couleur sera perçue différemment en fonction de la couleur de fond. D'autre part une proportion non négligeable de la population est daltonienne (environ 10%). Du côté informatique, les écrans ne sont généralement pas calibrés et une même couleur affichée sur deux écrans donne des résultats très différents.

L'œil est capable de percevoir le *mouvement*, grâce à des récepteurs spécialisés et grâce à la fusion entre percepts successifs effectuée au niveau cérébral. Que ce soit au cinéma, à la télévision ou sur un écran d'ordinateur, le mouvement est produit par l'affichage successif d'images fixes et exploite donc la fusion sensorielle. On fait référence en général au phénomène de persistance rétinienne pour expliquer cette perception, c'est-à-dire au fait que les cellules de la rétine conservent l'information pendant un temps de l'ordre de 1/20ème de seconde. Cependant l'œil peut capter des mouvements bien plus rapides (jusqu'à 1/100ème de seconde), et percevoir des mouvements dès une fréquence de l'ordre de 1/10ème de seconde. Dans tous les cas, un mouvement n'est perçu que si les images successives sont cohérentes : un diaporama projeté à 25 images par seconde ne sera pas perçu comme une animation. S'il suffit de 10 images par seconde pour produire la sensation de mouvement, la qualité de cette perception augmente avec la fréquence d'affichage. Ainsi, au cinéma (25 images/seconde), un panoramique donne une impression de flou. Le même panoramique filmé à 60 images/seconde (standard de cinéma ShowScan) donnera l'impression d'une image nette. Ainsi, la résolution temporelle est traduite en une résolution spatiale.

La perception de la *profondeur* (perception 3D) est possible par la combinaison de la vision stéréoscopique et de la vision active : à courte distance, les différences entre les images perçues par les deux yeux sont interprétées au niveau cérébral comme une information de profondeur. Au-delà de quelques mètres, c'est grâce aux variations perçues en bougeant la tête et le corps que les positions relatives des objets sont interprétées, en combinaison avec des caractéristiques apprises telles que l'effet de perspective (la taille apparente est plus petite lorsqu'un objet est plus loin). Les ordinateurs actuels sont capables de restituer des images stéréoscopiques (en imposant le port de lunettes spéciales à l'utilisateur). Pour que la sensation de 3D soit réaliste, il faut compléter cet équipement par un capteur de position de la tête qui permette de reproduire l'utilisation de la vision active.

Ouïe

L'ouïe a été moins étudiée que la vision, et beaucoup de phénomènes observés restent mal compris. Au plan physiologique, le système auditif est très perfectionné. La partie mécanique de l'oreille capte les vibrations de l'air. Sa bande passante est impressionnante, puisque le rapport entre les variations les plus faibles et les plus importantes qu'elle peut percevoir sans détérioration est de l'ordre d'un million. Aucun système mécanique n'est capable de telles performances.

Contrairement à la vue, on peut entendre sans écouter. En réalité, notre ouïe est toujours active : les signaux sonores sont constamment interprétés, même si un grand nombre ne parviennent pas à notre conscience. Nous sommes également capables de focaliser notre ouïe vers un point de l'espace, un phénomène appelé "effet cocktail party" : dans le brouhaha d'une réception, on peut concentrer son attention sur une discussion distante ou une autre. Ce phénomène est pour l'instant mal expliqué au plan psychologique.

Lorsque plusieurs sources sonores sont actives, des effets de masquage peuvent tromper notre perception et amener à confondre certaines d'entre elles. Pour être séparées, les sources doivent être à une distance suffisante, ou alors être dans des bandes de fréquences suffisamment différentes, etc. Les conditions exactes de masquage sont mal connues et dépendent beaucoup du contexte. Elle dépendent également des informations perçues par nos autres sens. Par exemple, lorsqu'une personne chante en play-back, nous avons l'impression que c'est le son de sa voix que l'on entend, à moins que l'information visuelle et l'information auditive soient trop contradictoires. Le même effet se produit avec les films doublés.

Nous sommes également capables de localiser une source sonore dans l'espace. Comme pour la vue, le fait d'avoir deux oreilles contribue à cette perception sans en être le seul vecteur. Lorsqu'une source est proche, le temps que met un train d'onde sonore pour parvenir aux deux oreilles est différent. Cette différence est "calculée" par le cerveau pour déterminer la direction de la source. Mais d'autres facteurs interviennent, qui deviennent prédominants lorsque les sources sont distantes : la tête et le haut du corps filtre le son provenant de la source, de telle sorte que les spectres fréquentiels des sons reçus aux deux oreilles sont différents ; de plus l'air absorbe certaines fréquences (surtout les fréquences aiguës) de telle sorte qu'un son distant est plus "étouffé" et moins fort qu'un son proche. C'est l'analyse de ces différences par le cerveau qui explique que l'on peut percevoir si une source est devant ou derrière, en haut ou en bas. En effet, la seule différence temporelle ne permet pas de désambiguïser ces cas.

C'est d'ailleurs en utilisant des têtes artificielles et en mettant des micros à la position des oreilles que l'on fait les enregistrements stéréophoniques les plus parfaits (à condition de les écouter ensuite avec un casque). Divers systèmes informatiques permettent de restituer du son 3D à travers un casque ou des haut-parleurs en reproduisant les décalages temporels et les filtres dus à la partie supérieure du corps humain. Ils sont utilisés dans des applications de réalité virtuelle et dans certains jeux.

Comme pour la vision, l'ouïe est un processus actif : pour localiser une source ou concentrer son attention, on bouge la tête afin de corréler les changements du signal perçu avec notre interprétation courante de la scène. Aussi, comme pour la vision stéréoscopique, l'effet de son 3D ne peut être réaliste que si l'on capte la position de la tête de l'utilisateur pour permettre une audition active.

Malgré ses capacités, l'ouïe est très peu sollicitée par les systèmes interactifs actuels, en dehors de "bips" plus ou moins variés et de signaux enregistrés. Certains travaux, restés au stade de la recherche, ont montré l'augmentation de performance, pour certaines tâches informatiques, qui peut résulter de l'usage du son, à la fois pour décharger le canal visuel, pour faciliter le suivi d'activités d'arrière-plan ou pour notifier l'utilisateur de l'occurrence d'événements asynchrones (arrivée de message par exemple).

Toucher

On peut associer au toucher trois composantes : le toucher, la proprioception et la kinesthésie.

Le toucher concerne la perception de la température, de la pression et de la douleur. C'est en particulier la perception de la pression qui nous permet d'évaluer la texture d'un objet. Les capteurs tactiles sont beaucoup plus denses dans certaines régions du corps (doigts et bouche notamment) que dans d'autres (dos par exemple). D'autre part ces capteurs sont sensibles aux variations de pression. Comme pour la vue et l'ouïe, c'est dans une situation de perception active que l'on peut le mieux exploiter ce sens tactile : pour évaluer la matière d'un objet (bois, plastique, papier, coton, peau, etc.), on touche l'objet en déplaçant ses doigts dessus. C'est dans les systèmes interactifs pour handicapés que l'on a le plus cherché à exploiter le sens tactile, et en particulier pour les aveugles et malvoyants. Divers dispositifs existent (mécaniques, piézo-électriques, pneumatiques, ...) permettant de restituer une information tactile. Le développement de ces périphériques bénéficierait également aux interfaces classiques. Par exemple, lorsque l'on déplace la souris, un retour d'information tactile pourrait nous renseigner sur la position du curseur (fenêtre, barre de menus, fond d'écran, etc.).

Une autre composante du toucher est notre capacité à connaître la configuration de notre corps dans l'espace : je sais, sans avoir à la regarder, si ma main est ouverte ou fermée, si je suis debout, accroupi ou allongé, etc. Combiné avec le sens tactile, ce sens dit "proprioceptif" nous permet notamment d'appréhender la forme des objets sans les voir. Les systèmes actuels de réalité virtuelle souffrent de l'absence d'utilisation de ce canal : lorsque l'on saisit un objet virtuel, aucune sensation tactile ni proprioceptive ne vient confirmer le contact et ne permet d'évaluer la forme, la texture, le poids de l'objet. (Ils existe cependant quelques prototypes de recherche qui explorent cette voie). Comme le toucher, la proprioception est active. Une expérience simple montre qu'il est impossible pour un sujet ayant les yeux bandés de reconnaître un objet qui est posé dans sa main dans différentes positions, alors que cette même tâche est très facile si l'on autorise le sujet à manipuler l'objet.

La dernière composante du sens tactile est la kinesthésie (qui, pour certains auteurs, englobe la proprioception). Le sens kinesthésique nous permet de connaître l'effort que font nos muscles par exemple lorsque l'on soulève un objet (poids) ou lorsque l'on pousse un objet (résistance). Comme la proprioception, l'exploitation de ce sens fait cruellement défaut aux systèmes actuels de réalité virtuelle. Il existe des dispositifs, comme le Clavier Rétroactif Modulaire de l'ACROE (Grenoble), qui permettent de restituer des sensations tactiles et kinesthésiques grâce à un contrôle en temps réel et à haute fréquence (plusieurs centaines de Hz) des touches d'un clavier de type piano. Un autre bon exemple est le pantographe du CRIM (Montréal) qui peut se substituer à une souris et qui permet également de restituer des sensations tactiles et de retour d'effort. Des dispositifs de ce type sont actuellement expérimentés dans le milieu médical pour permettre à un médecin non seulement d'observer une tumeur par imagerie médicale (reconstruction d'images 3D à partir de scanners, tomographies et RMN) mais également de la palper.

Le couplage action-perception

Comme nous l'avons vu, la perception est presque toujours un phénomène actif, et l'on ne peut complètement séparer la perception de l'action : nos actions sont constamment régulées par notre perception et notre perception dirige constamment nos actions. Lorsque par exemple on saisit un objet, la trajectoire de la main est constamment réajustée en fonction de la perception d'abord visuelle puis tactile (lorsqu'il y a contact). On parle de *boucle perception-action.*

Selon certains psychologues (l'approche dite "écologique" de Gibson), l'être humain ne perçoit pas des unités d'information (image, son, pression, etc.) mais il perçoit des flux sensoriels. C'est l'extraction des invariants de ces flux qui lui permet d'appréhender le monde qui l'entoure, qu'il s'agisse de flux visuels, auditifs, tactiles, gustatifs ou olfactifs. La perception active consiste donc à agir sur ces flux de façon contrôlée (par exemple en déplaçant la tête), afin de faciliter l'extraction des invariants.

L'importance de cette boucle action-perception peut être illustrée par une expérience de Held et Hein : deux chats sont élevés dans le noir complet jusqu'à l'âge de dix semaines. L'un des chats est dit actif, l'autre passif. Ensuite, à raison de trois heures par jour, ils sont mis dans un dispositif tel que tout déplacement du chat actif provoque un déplacement identique du chat passif (le chat passif ne peut se déplacer seul). L'expérience montre que le développement du chat actif est normal, tandis que le chat passif n'a aucune perception de la profondeur. (Heureusement, cette perception apparaît si le chat passif est rendu actif pendant 2 jours.)

Sur le plan des systèmes interactifs, de nombreuses expérimentations (moins cruelles pour les sujets que celle de Held et Hein !) ont montré que les techniques d'interaction qui exploitent la boucle perception-action sont plus performantes que les techniques "passives". Malheureusement, ce message n'est pas encore passé dans l'industrie et la plupart des systèmes interactifs commerciaux sont "passifs".

Action

L'homme a deux moyens d'action immédiate sur le monde qui l'entoure : l'action physique et le langage parlé.

Action physique

L'action physique utilise le système moteur pour agir sur des objets de notre environnement. Chez l'homme, la main est l'organe le plus sophistiqué pour l'action physique. L'action manuelle, qui est le moyen d'action unique des systèmes interactifs actuels (à de rares exceptions près), ne peut être découplée de la perception, comme nous l'avons vu.

D'une part, c'est dans la main que les récepteurs tactiles sont les plus nombreux. Toute action manuelle sera donc exécutée avec un retour permanent d'information tactile et kinesthésique, même pour un périphérique "passif". Ainsi, bien que, du point de vue de la machine, une souris, un trackball, un joystick ou une tablette graphique fournissent tous une position 2D et sont donc équivalents, il n'en va pas de même pour l'utilisateur : il est par exemple bien plus difficile de signer avec une souris qu'avec un stylet sur une tablette, et c'est pratiquement impossible avec un joystick.

D'autre part, beaucoup d'actions manuelles sont guidées par la vue : lorsque l'on attrape un objet, le geste est ajusté en temps réel grâce à l'information visuelle qui indique la position relative de la main et de l'objet. Ce guidage peut même se faire si l'objet est dans le champ de vision périphérique : lorsque l'on utilise un clavier et une souris, la main va de l'un à l'autre alors que le regard reste fixé sur l'écran et pourtant la main est guidée par la vue.

Dans les tâches d'interaction manuelle, deux actions sont très fréquentes : la saisie d'un objet, et la manipulation d'un objet. La saisie a été étudiée empiriquement afin de déterminer s'il existait une loi permettant d'évaluer le temps d'une saisie. Paul Fitts a découvert une loi empirique pour les tâches de pointage. Il s'agit de pointer, avec le doigt, une cible circulaire de taille L à une distance D. Intuitivement, le temps de pointage doit augmenter si D augmente et si L diminue : le temps de pointage est plus long si la cible est plus éloignée ou si elle est plus petite. La loi de Fitts s'exprime sous la forme suivante :

$$t = 0.1 \log 2D/L$$

Le temps t (en seconde) de pointage d'une cible de taille L à une distance D est proportionnel au logarithme de 2D/L. Ainsi, si la distance double ou si la taille de la cible diminue de moitié, le temps de pointage ne double pas mais croît selon une loi logarithmique. Cette loi empirique s'applique également au pointage avec un dispositif tel que souris, joystick ou tablette. A titre d'exemple, voici les temps estimés par la Loi de Fitts pour des tâches de pointage courantes dans une interface graphique. La première ligne correspond au pointage d'un icone dans une fenêtre, la seconde au pointage d'un trait dans un logiciel de dessin et la troisième à la sélection d'un menu dans la barre de menus :

D	L	temps
10 cm	1 cm	0,4 s
10 cm	1 mm	0,8 s
30 cm	0,5 cm	0,7 s

La loi de Fitts est l'une des très rares lois quantitatives que nous fournit la psychologie pour la réalisation de systèmes interactifs. Elle montre qu'un temps de pointage typique est compris entre 1/2 et 1 seconde. Si l'on accumule les temps de pointage sur une longue période d'interaction, on découvre que tout gain sur ces temps de pointage est appréciable. Dans les logiciels actuels, l'utilisation d'équivalents-claviers et accélérateurs divers est souvent un moyen d'éviter des pointages longs et il est caractéristique d'observer que les utilisateurs experts, qui optimisent l'utilisation du logiciel, exploitent au maximum ces raccourcis. Nous verrons que certaines techniques d'interaction peuvent également faire gagner du temps en diminuant le nombre ou le degré de difficulté des pointages, sans recourir à des raccourcis que l'utilisateur doit mémoriser.

Dans la vie quotidienne, la plupart des actions manuelles que nous effectuons emploient les deux mains. La psychologie classique a étudié l'usage des deux mains sous l'angle de la préférence manuelle, en considérant que "la main gauche est une mauvaise main droite" (ou l'inverse chez les gauchers). Cette vision est réductrice car en réalité les deux mains coopèrent, avec des rôles distincts. Ainsi, lorsque l'on écrit sur une feuille de papier, la main gauche déplace la feuille afin que la main droite soit dans la bonne position pour écrire. Le psychologue français Yves Guiard a montré que dans la majorité des tâche bimanuelles, la main non-dominante (gauche pour un droitier, droite pour un gaucher) intervient avant la main dominante et qu'elle situe le contexte dans lequel agit la main dominante.

Les systèmes interactifs actuels n'exploitent pas l'interaction bimanuelle : la frappe au clavier est certes bimanuelle mais les deux mains ont un rôle parfaitement symétrique. Par contre, les interfaces sont munies d'un seul dispositif de désignation (généralement une souris), manipulé de la main dominante. Une forme primitive d'interaction bimanuelle apparaît cependant dans certains cas, lorsque la main non-dominante sélectionne au clavier, par des raccourcis ou des touches de fonction, un mode qui détermine les actions que peut réaliser la main dominante par l'intermédiaire de la souris. Par exemple, dans un simulateur de vol, la main droite pilote tandis que la main gauche sélectionne la vue affichée à l'écran.

Des systèmes expérimentaux utilisent un trackball pour la main gauche en plus de la traditionnelle souris pour la main droite. Ces systèmes ont permis de développer des techniques d'interaction véritablement bimanuelles (que nous présenterons plus loin) plus performantes que les techniques classiques. Par exemple, la main gauche permet de contrôler le défilement d'un document pendant que la main droite le modifie.

Langage

L'étude du langage est un volet très important de la psychologie (complété par la linguistique qui s'intéresse plutôt aux langues). En informatique, les langages de programmation reprennent certaines caractéristiques des langages humains, et les interfaces conversationnelles (voir l'historique des styles d'interaction ci-dessous) sont fondées sur des langages de commandes qui sont des langages de programmation simplifiés. D'autre part, on a depuis longtemps développé des systèmes dont l'objectif est de permettre une communication en langue naturelle entre l'utilisateur et le système informatique.

L'utilisation d'un langage (parlé, écrit, gestuel, voire olfactif chez certaines espèces) suppose que les acteurs de la communication partagent ce langage. Chez l'homme, l'apprentissage d'une langue dure plusieurs années, et est en grande partie indissociable de l'acquisition d'une grande quantité de traits sociaux et culturels et de ce que l'on appelle "le sens commun". Compte tenu de la complexité des langues humaines et de l'imbrication entre langage et environnement social et culturel, et malgré l'augmentation de la puissance de calcul des ordinateurs, les interfaces en langue naturelle sont restées confinées à des domaines d'application spécialisés, dans lesquels on peut simplifier le vocabulaire, la syntaxe et le domaine du discours de façon suffisamment importante pour pouvoir donner au système informatique le rôle d'un interlocuteur.

Dans la suite de ce cours, nous nous intéressons essentiellement aux systèmes utilisant l'interaction graphique. C'est pourquoi nous ne développons pas plus avant les acquis de la psychologie dans le domaine du langage, de même que nous n'étudierons pas les méthodes et techniques d'interaction en langue naturelle.

Cognition

Le système cognitif humain est le plus complexe du règne animal. Sans prétendre faire le tour des connaissances en psychologie cognitive (loin de là !), on peut identifier trois catégories de traitements réalisés par le cerveau humain : le traitement de l'information sensorielle, que nous avons intégré aux sections précédentes, la mémoire et l'apprentissage, et la résolution de problèmes.

Mémoire et apprentissage

Les théories les plus courantes de la psychologie considèrent qu'il existe trois types de mémoire : la mémoire sensorielle, la mémoire à court terme et la mémoire à long terme. La mémoire sensorielle est associée aux sens (vue, ouïe, etc.) et stocke l'information sensorielle pendant un court laps de temps (de l'ordre d'1/4 de seconde), probablement afin de l'encoder avant qu'elle ne soit stockée dans la mémoire à court terme.

La *mémoire à court terme* permet de stocker un nombre limité de mnèmes (unités d'information de la mémoire : images, lettres, sons, etc.) pendant une durée de quelques dizaines de secondes au maximum. Ainsi, lorsque l'on lit un numéro de téléphone dans un annuaire, on le mémorise suffisamment pour pouvoir le composer mais il est improbable que l'on se rappelle du numéro un demi-heure après, ou même 5 minutes après. La capacité de la mémoire à court terme est très faible, de l'ordre de 7 mnèmes. Si un sujet lit une liste de 7 noms et qu'on lui demande si un nom est dans la liste, il y a toutes les chances qu'il donne une réponse correcte. Si la liste est plus longue, le taux d'erreur augmente très vite.

Le mémoire à court terme peut être vue comme une mémoire de travail. Sa faible capacité doit être prise en compte lorsque l'on conçoit un système interactif : par exemple, il est préférable de limiter le nombre d'items dans un menu à 7 environ, et il faut éviter d'obliger l'utilisateur à garder en mémoire des informations sur l'état du système (comme par exemple le mode courant), qui mobilisent inutilement sa mémoire à court terme.

Lorsque l'on a besoin de mémoriser une information pendant une durée supérieure à celle du stockage dans la mémoire à court terme ou lorsque l'on doit mémoriser plus de mnèmes que la capacité de la mémoire à court terme, on emploie la répétition. Ainsi, si le numéro que l'on a trouvé dans l'annuaire sonne occupé, on le répète (à voix haute ou dans sa tête) pour être sûr "de ne pas le perdre". Lorsque l'on utilise un numéro fréquemment, on finit par "le connaître", c'est-à-dire qu'il est passé dans la mémoire à long terme.

La *mémoire à long terme* est très mal connue. On considère que sa capacité comme son temps de stockage sont infinis. L'un des mécanismes pour que des informations passent dans la mémoire à long terme est la répétition, comme on vient de le voir. Mais ce n'est pas le seul : certains événements, qui ne se sont produits qu'une fois, sont gravés à jamais dans notre mémoire. Même des informations apparemment plus anodines sont stockées dans la mémoire à long terme. Par exemple, on reconnaît très souvent un film ou une image que l'on a déjà vu (sans forcément se souvenir du lieu et de la date où on l'a vu). De plus, on sait également reconnaître instantanément des choses que l'on n'a pas en mémoire (on sait ce que l'on ne sait pas), par exemple un visage ou un mot.

Ce qui limite l'usage de la mémoire à long terme, c'est l'accès aux informations. Il est fréquent de savoir que l'on connaît un nom ou un téléphone, mais de ne pas arriver à le retrouver ... sauf lorsque l'on en n'a plus besoin ! Ou bien, lorsque l'on voit un film pour la seconde fois, on se souvient de ce qui va se produire au fur et à mesure que l'on revoit le film, sans pour autant être capable de se rappeler de toute l'intrigue. Ainsi, la mémoire à long terme semble fonctionner sur un mode associatif : une scène du film nous permet de nous rappeler la suite immédiate, mais pas plus. Les systèmes mnémotechniques servent d'ailleurs à construire des associations qui facilitent l'accès à certaines informations, en exploitant le fait que certains types d'informations (par exemple des mots) sont plus faciles à mémoriser que d'autres (par exemple des séquences de chiffres).

L'impact de la mémoire à long terme sur la conception de systèmes interactifs est lié à la répétition : un utilisateur régulier d'un logiciel mémorisera progressivement la liste des commandes des menus qu'il utilise le plus souvent. En présentant les équivalents-clavier de ces commandes dans les menus eux-mêmes, l'utilisateur peut apprendre les raccourcis qui lui feront gagner du temps. Ainsi, l'interface doit être adaptée à plusieurs catégories d'utilisateurs, que l'on sépare souvent en utilisateurs novices, occasionnels et experts. La principale différence entre ces catégories est la quantité d'information concernant le logiciel qui est stockée dans leur mémoire à long terme et la facilité d'accès à ces informations.

Résolution de problèmes

Si le terme "résolution de problèmes" évoque une activité intellectuelle intense, il désigne en psychologie toute activité visant à atteindre un but à partir d'une situation courante. Chercher où l'on a mis sa montre procède de la résolution de problème au même titre que résoudre un casse-tête, décider ce que l'on va manger ce soir ou démontrer un théorème.

De très nombreux travaux ont étudié les stratégies de résolution de problème utilisées par les humains sans que l'on ait aujourd'hui de théorie définitive. (Certains résultats de ces travaux ont été utilisés par l'Intelligence Artificielle). Il s'avère que l'usage de la logique (au sens de la logique mathématique) est assez rare dans la résolution de problème, et que lorsque c'est le cas les logiques utilisées sont des approximations de la logique mathématique. Bien plus fréquent est l'utilisation de *règles heuristiques*, c'est-à-dire de règles issues de l'expérience de chacun, et qui consistent à simplifier le problème en problèmes plus simples ou que l'on sait déjà résoudre, même si la solution finale n'est pas optimale. Ces règles heuristiques sont basées sur des *modèles mentaux*, représentations mentales abstraites de situations qui peuvent servir de modèles pour la résolution de problèmes similaires.

Une façon (simplifiée) de décrire le processus de résolution de problème est la suivante : on cherche à atteindre un état final F à partir de l'état de départ D. Si l'on ne sait pas atteindre F par une action simple, on décompose le problème en sous-problèmes, c'est-à-dire que l'on se fixe un but intermédiaire I, que l'on cherche à atteindre depuis D. Supposons que l'on pense atteindre I depuis D par l'action A. On exécute l'action A, et l'on compare l'état obtenu Ir à l'état attendu I. S'il y a identité entre Ir et I, on peut poursuivre la résolution en essayant d'atteindre F à partir de I. Si Ir est différent de I, on peut essayer de défaire l'action A pour revenir à l'état initial D, ou l'on peut essayer de réparer l'erreur pour aller de Ir à I, ou enfin on peut se satisfaire de cet état et essayer d'aller à F depuis Ir (Figure 4).

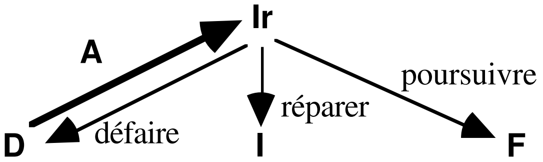


Figure 4 : stratégies lorsque le but atteint Ir n'est pas le but attendu I

La Théorie de l'Action de Don Norman précise le déroulement du processus mental lors d'une étape élémentaire de la résolution de problème, dans le cas de l'interaction avec un système, informatique ou autre (Figure 5).

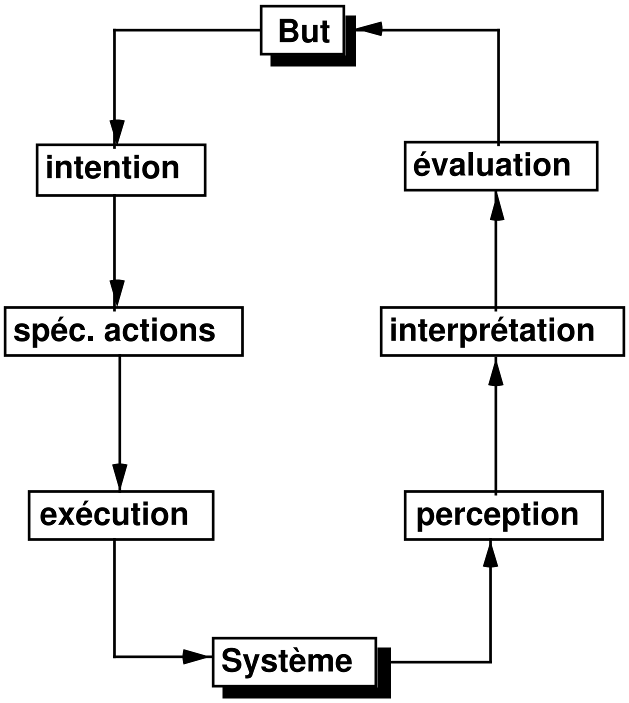


Figure 5 : Théorie de l'Action (Don Norman)

A partir du but que l'utilisateur s'est défini, la première étape est l'intention, c'est-à-dire le choix de la méthode d'action ; puis vient la spécification des actions (physiques ou langagières) à effectuer pour concrétiser cette intention et enfin l'exécution de ces actions. Les actions de l'utilisateur sont alors interprétées par le système comme des commandes et le traitement de ces commandes produit un changement d'état perceptible par l'utilisateur. Notons que l'absence de changement d'état est également perçue par l'utilisateur : l'absence de réponse est en soi une réponse. Après avoir perçu la réponse (ou la non-réponse) du système, l'utilisateur interprète le résultat (en fonction de son modèle mental du système), et enfin l'évalue par rapport au but qu'il s'était fixé. Si le but est atteint, l'utilisateur peut passer au sous-but suivant, sinon il doit défaire, réparer ou poursuivre comme expliqué plus haut.

L'avantage de cette description de l'activité de l'utilisateur est de mettre en évidence ce que Norman appelle des distances : *distance d'exécution* entre le But et le Système dans la branche descendante et *distance d'évaluation* entre le Système et le But dans la branche ascendante. L'évaluation de ces distances pour un système donné permet d'avoir une idée de sa difficulté d'utilisation. Supposons par exemple que l'utilisateur ait détruit un fichier avec un shell Unix. La réponse du système est de lui rendre la main, à moins qu'une erreur ne se soit produite. L'utilisateur doit donc interpréter l'apparition du "prompt" comme le fait que la commande s'est bien déroulée et que le fichier a bien été détruit. Beaucoup d'utilisateurs, même experts, préfèrent s'en assurer en exécutant la commande "ls" et en vérifiant que le fichier a bien disparu, c'est-à-dire en exécutant un nouveau cycle dont le but est "vérifier que le fichier a bien été détruit". Cela traduit à l'évidence le fait que la distance d'évaluation est trop grande. Dans une interface iconique comme le Finder du Macintosh, lorsque l'on détruit un fichier en l'amenant dans la poubelle, celle-ci change de forme, et l'icone du fichier a bel et bien disparu : la distance d'évaluation est plus faible.

La Théorie de l'Action ne doit pas faire croire que l'utilisateur planifie ses actions longtemps à l'avance. En réalité, l'activité humaine s'adapte constamment au contexte de son exécution et à son propre déroulement : on parle d'*action en situation* ("situated action"). La résolution de problème est donc un processus incrémental et non pas une activité planifiée. Même dans les activités qui semblent extrêmement codifiées, comme par exemple le pilotage d'un avion, le contrôle du trafic aérien, la surveillance d'une centrale nucléaire, la construction d'un satellite, etc., où des milliers de pages de documentation décrivent toutes les procédures pas à pas, on constate que le comportement des usagers, surtout dans des situations non standard (mais prévues), est très variable et se conforme rarement aux procédures.

L'utilisation d'un système informatique suit la même règle : chaque utilisateur, en fonction de son environnement, de ses habitudes, des modèles mentaux qu'il s'est construits en utilisant le système, mais aussi d'autres systèmes, invente ses propres méthodes de résolution de problème, avec le plus souvent pour seul guide l'efficacité immédiate plutôt que la compréhension profonde du fonctionnement réel du système. Cette caractéristique doit être prise en compte lors du développement du système, notamment en adaptant celui-ci aux utilisateurs visés, en conduisant des études d'utilisabilité avec des utilisateurs réels, et même si possible en impliquant les utilisateurs dès la conception du système. Si les utilisateurs sont par nature imprévisibles (et c'est la raison pour laquelle les systèmes interactifs sont des systèmes ouverts), il sont aussi beaucoup plus susceptibles d'adopter un système s'ils ont participé à sa mise en œuvre (comme le montrent divers travaux de psychologie sociale).

Historique de l'Interaction Homme-Machine

L'histoire de l'interaction homme-machine est presque aussi vieille (ou jeune !) que l'histoire de l'informatique. En 1945, Vannevar Bush, dans son célèbre article "As We May Think" décrivait un système électronique (imaginaire) permettant le stockage et la recherche d'informations, et inventait les concepts de l'hypertexte : navigation, indexation, annotation. Au début des années 60, alors que les systèmes à temps partagé font leurs débuts, Ivan Sutherland crée le système SketchPad (1963), qui utilise comme écran un oscilloscope. SketchPad est un outil de dessin qui permet de créer des schémas par instanciation de modèles prédéfinis (à l'instar du modèle classe instance des langages à objets - Simula date de la même époque). SketchPad permet la construction de structures hiérarchiques et utilise des contraintes pour définir les objets. L'interaction utilise un crayon optique. A la fin des années 60, Douglas Engelbart développe le système NLS-Augment. En 1968, il présente une démonstration à San Francisco où l'on voit un système de traitement de texte hiérarchique, multimédia (texte et graphique) et hypertexte. C'est un système permettant le travail coopératif : on peut partager des fichiers, annoter des documents, utiliser une messagerie, et même un système de visio-conférence avec partage d'écrans informatiques et télépointeurs. Engelbart a inventé la souris (et a déposé un brevet qui ne lui rapportera rien puisque les souris qui seront commercialisées utiliseront un principe différent de son brevet), ainsi que le clavier à une main, les fenêtres, etc.

En 1969, Alan Kay, alors chez Xerox et aujourd'hui Apple Fellow, présente sa vision d'un ordinateur de poche qui préfigure les "PDAs" (Personal Digital Assistants). Dans les années 70 et 80 (en encore aujourd'hui), les laboratoires de Xerox PARC (Xerox Palo Alto Research Center) sont un creuset sans équivalent pour le développement de systèmes interactifs novateurs. Pendant les années 70, le projet Smalltalk, qui lancera les langages à objets, conduit à la réalisation de la première station de travail à écran graphique et souris, l'Alto, qui dispose également d'un réseau local (Ethernet a été inventé aussi à Xerox PARC). L'interface contient déjà des fenêtres, menus, barres de défilement et utilise la sélection directe à la souris. En 1981, une nouvelle machine, le Star, utilise la métaphore du bureau avec des icones représentant le système de fichiers et un certain nombre de commandes génériques. Le Star invente aussi la présentation WYSIWYG (What You See Is What You Get) : les documents apparaissent à l'écran sous la même forme que lorsqu'ils sont imprimés. Le Star sera un échec commercial (comme beaucoup de tentatives de Xerox de commercialiser ses inventions dans le domaine de l'informatique) mais inspirera Apple : en 1983, l'Apple Lisa est une copie du Star mais, trop cher et trop étriqué, c'est aussi un échec commercial. En 1984, le Macintosh, une évolution du Lisa vendu à un prix plus bas est un succès commercial sans précédent. Il faudra plus de 10 ans pour que Microsoft intègre à son interface graphique Windows des fonctionnalités comme l'interaction iconique ou le cliquer-tirer.

Dans la seconde moitié des années 80, au Massachusetts Institute of Technology (MIT) à Boston, un grand projet financé notamment par Digital Equipment Corp. (DEC), IBM et Tektronix a pour objectif de fournir à tous les étudiants du campus des postes de travail accessibles librement. C'est le projet Athena, qui donnera naissance notamment au X Window System, qui deviendra l'interface graphique standard des stations Unix. A la fin des années 80, on voit apparaître les premiers systèmes de réalité virtuelle qui immergent l'utilisateur dans un monde synthétique. Au début des années 90, les physiciens du CERN inventent un système hypertexte qui s'appellera plus tard le World Wide Web et envahira la planète.

Malgré cette profusion d'inventions, qu'il faudrait compléter par tous les travaux, moins spectaculaires mais indispensables, de nombreux chercheurs et praticiens, il faut reconnaître que les systèmes interactifs actuels sont souvent loin des attentes des utilisateurs. Des études ont montré que l'utilisation de l'informatique n'a pas augmenté la productivité des "cols blancs", et il n'est pas besoin d'étude pour constater que l'informatique n'a pas réduit la quantité de papier que l'on est amené à manipuler, mais au contraire l'a fait augmenter. C'est la preuve que la conception de systèmes interactifs de qualité est encore un domaine mal maîtrisé et souvent sous-estimé. On pourrait espérer que cette situation soit en train de changer et que la qualité de l'interface devienne un argument de vente de plus en plus important, grâce au développement de la concurrence sur un marché en pleine expansion. Malheureusement, lorsque la norme est la médiocrité et lorsque le marché est dominé par des acteurs qui ont la mainmise sur les technologies-clés de l'interaction, on ne peut être trop optimiste.

Historique des styles d'interaction

On peut classer la majeure partie des systèmes interactifs en trois catégories selon le style d'interaction qu'ils emploient. Le terme de "style d'interaction" recouvre l'ensemble des méthodes, techniques et règles d'interaction qui sont employées dans un système (cette définition est volontairement vague).

Systèmes conversationnels

Historiquement, le style conversationnel est le plus ancien : il correspond à un mode d'interaction langagière inspiré de la conversation entre humains. Le langage humain a également inspiré les langages de programmation, il n'est donc pas surprenant de le retrouver dans l'interaction.

Dans un système conversationnel, les commandes sont fournies en tapant un texte au clavier, généralement une ligne terminée par un retour-chariot. Un feed-back permet à l'utilisateur de voir ce qu'il tape (et de le corriger), mais la commande n'est exécutée que lors de sa validation. A ce moment-là, la commande est interprétée et transformée en opérations sur les objets internes du systèmes (voir le modèle conceptuel générique de la figure 2). Le système fournit alors une réponse qui est soit un message d'erreur, soit le résultat, éventuellement vide, de la commande. L'utilisateur peut alors entrer une nouvelle commande, etc. Les shells de commandes comme ceux d'Unix sont des exemples de systèmes conversationnels.

L'inconvénient principal d'un système conversationnel est que l'utilisateur doit apprendre le langage de commande et que les réponses du système sont parfois difficile à interpréter. Il en résulte une distance d'exécution et une distance d'évaluation importantes (voir la Théorie de l'Action). Par contre, pour des utilisateurs experts, ce mode d'interaction peut être plus efficace que tout autre, notamment par la possibilité d'étendre le système en programmant ses propres commandes (comme par exemple les shell-scripts sous Unix). D'autre part, les systèmes conversationnels impliquent un séquencement strict entre actions de l'utilisateur et réponses du systèmes (à l'exception du fait que l'utilisateur peut interrompre le système si celui-ci tarde à répondre). Il est donc difficile de mener plusieurs tâches simultanément, alors que c'est une activité courante dans la vie quotidienne.

Menus, formulaires

Les systèmes à base de menus et de formulaires sont une extension du modèle conversationnel destinée à faciliter l'interaction en rendant le langage de commande explicite. Ainsi, la sélection d'une commande dans un menu nécessite de reconnaître cette commande dans une liste, alors qu'une interface conversationnelle oblige à se souvenir du nom de la commande. Les menus hiérarchiques permettent d'accéder à un grand nombre de commandes, au prix de commandes de navigation qui permettent de passer d'un menu à l'autre.

La syntaxe disponible avec un simple système de menus est très primitive. C'est pourquoi ce style d'interaction est complété par l'usage de formulaires, similaires dans leur principe aux formulaires papier : lorsque le formulaire lui est présenté à l'écran, l'utilisateur doit remplir ses champs (dans un ordre quelconque), puis valider le formulaire. C'est la validation qui déclenche l'exécution d'une commande par le système, et sa réponse sous forme d'un message d'erreur, d'un nouveau formulaire, d'un menu ou d'un document. Dans ce dernier cas, des commandes de navigation permettent de parcourir un document qui ne peut être totalement affiché à l'écran (dans le cas d'interfaces graphiques). La plupart des services Minitel sont des systèmes à base de menus et de formulaires, de même que beaucoup d'applications de gestion (au sens large du terme : voir par exemple les systèmes de réservation d'avions ou de trains).

Les systèmes à base de menus et de formulaires imposent, comme les systèmes conversationnels, un dialogue strict entre l'utilisateur et le système. Cependant, ces systèmes assistent l'utilisateur en lui fournissant des informations permettant de guider ses actions : liste des commandes disponibles dans les menus, noms des champs dans les formulaires, etc. De plus l'interaction avec un formulaire n'impose pas d'ordre de remplissage et en cas d'erreur, il n'est en général pas nécessaire de tout re-saisir. Le fait de présenter les objets informatiques sous une forme graphique et la possibilité d'agir directement sur ces objets (cliquer dans un champ pour le remplir, sur un bouton OK pour valider) renforce la sensation d'engagement de l'utilisateur. L'inconvénient de ce style d'interaction est qu'il ne peut convenir qu'à des tâches prédéfinies. Comme pour le style conversationnel, il n'est en général pas possible de traiter plusieurs tâches en parallèle ou même hiérarchiquement. Par exemple, si l'un des champs à saisir est un numéro de client et que l'on ne connaît que son nom, il faudrait pouvoir utiliser le formulaire d'accès à la base de données des clients, sans avoir à abandonner le formulaire en cours. Enfin, les systèmes à base de menus et de formulaires ne sont en général pas programmables par l'utilisateur : si l'on utilise souvent le même formulaire avec les mêmes valeurs de champs, il est probable que l'on doive les entrer à chaque fois.

Navigation (hypertexte)

Un système hypertexte permet la navigation dans un ensemble de *nœuds*, reliés entre eux par un ensemble de *liens*. Un nœud se présente typiquement comme une page à l'écran. Les liens peuvent apparaître dans le corps de la page (mots soulignés ou encadrés) et/ou dans des menus/palettes. L'interaction consiste simplement à sélectionner des liens. D'autres primitives de navigation complètent les liens : navigation avant et arrière dans les nœuds visités récemment, navigation directe vers un nœud par son nom, etc.

Des exemples de systèmes de navigation sont les navigateurs du World Wide Web et le système d'information d'Emacs. On trouve également des systèmes de navigation comme sous-systèmes d'autres systèmes interactifs, notamment les systèmes d'aide en ligne.

Manipulation directe

L'avènement des stations de travail à écran graphique a amené les concepteurs d'interfaces à rapprocher l'interaction avec les objets informatiques de l'interaction avec les objets physiques. Ainsi, on a cherché à présenter les documents à l'écran sous une forme comparable à leur forme imprimée, tout en permettant de modifier ces documents en agissant directement dessus. Le développement de la métaphore du bureau procède de la même logique : en représentant des objets informatiques tels que fichiers et dossiers par des images évocatrices (les icones), et en permettant l'action sur ces icones par l'intermédiaire de la souris, on espère rendre les commandes possibles suffisamment intuitives pour éviter à l'utilisateur un apprentissage long et fastidieux.

Le terme de manipulation directe a été inventée par Ben Shneiderman en 1983, inspiré par certains logiciels de l'époque, notamment les premiers tableurs et les jeux vidéo. Shneiderman caractérise les applications à manipulation directe par 4 principes :

- affichage permanent des objets d'intérêt (les objets d'intérêt sont les objets qui ont un sens pour l'utilisateur, qui appartiennent à son modèle mental : par exemple les mots, lignes et paragraphes d'un texte) ;
- action directe (via un dispositif de désignation comme la souris) sur les objets d'intérêt plutôt que syntaxe complexe ;
- actions incrémentales et réversibles dont l'effet sur les objets d'intérêt est immédiatement visible ;
- structuration de l'interface en couches afin de faciliter l'apprentissage.

Comparé aux styles d'interaction précédents, la manipulation directe permet de traiter des tâches non prédéfinies, en particulier les tâches créatives : édition de texte, de dessins, de schémas, de partitions, etc. L'invention des systèmes de multi-fenêtrage, qui a accompagné celle de la manipulation directe, permet de mener de front plusieurs tâches, de façon entrelacée. La manipulation directe permet non seulement d'interagir avec plusieurs applications, mais aussi de transférer des données entre applications (par cliquer-tirer ou couper-coller, notamment). L'inconvénient principal de la manipulation directe est de ne pas permettre aisément la programmation par l'utilisateur : si l'on doit copier tous les fichiers qui ont un nom se terminant par ".c", il est probable que l'on doive faire cette opération "à la main".

Un autre problème, qui n'est pas inhérent au style d'interaction lui-même, est que de nombreuses applications n'exploitent pas la manipulation directe correctement : beaucoup de commandes sont déclenchées par la combinaison d'une commande de menu et d'une (ou plusieurs) boîtes de dialogue. Il n'y a alors ni action directe sur les objets d'intérêt (la manipulation se fait dans la boîte de dialogue), ni effet immédiat de l'action (il faut valider pour voir l'effet). On peut alors parler de manipulation indirecte (proche en réalité du style menus/formulaires), et non de manipulation directe. Nous verrons que la raison de cet état de fait tient en grande partie aux outils de développement aujourd'hui disponibles, qui simplifient le développement des parties de l'interface utilisant la manipulation indirecte sans apporter de solution satisfaisante pour mettre en œuvre les principes de la manipulation directe.

Deux catégories très répandues d'interfaces à manipulation directe sont l'édition de documents et l'interaction iconique.

Edition de documents

Le principe du WYSIWYG (What You See Is What You Get) est de présenter un document à l'écran sous une forme la plus proche possible de sa forme imprimée : texte, schéma, partition musicale, etc. Ainsi, dans un système de traitement de texte WYSIWIG, l'équation

$$d = \sqrt{b^2 - 4ac}$$

apparaît telle quelle, tandis que dans un système non WYSIWYG, elle aurait par exemple la forme

`$d \equal \sqr{b^2 - 4 a c}$`

L'interaction avec un document WYSIWYG utilise en général

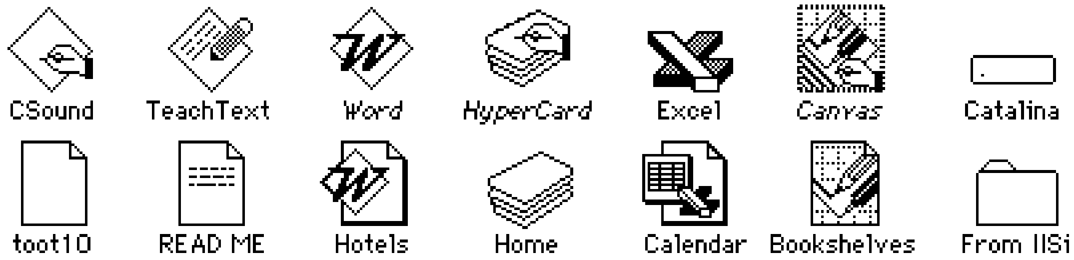
- des techniques de manipulation directe ;
- des techniques de manipulation "indirecte" (boîtes de dialogues, menus, palettes)

Pour mettre en œuvre les techniques de manipulation directe, il peut être nécessaire de violer le principe de WYSIWYG. Par exemple, dans un logiciel comme Word, on peut visualiser le texte en mode pleine page ou en mode défilement, on peut afficher les caractères normalement invisibles (tabulations, espaces, retour-chariot, etc.).

Cependant, la notion de WYSIWYG est indépendante de celle de manipulation directe. On peut faire de la manipulation directe de représentations non WYSIWYG, et on peut éditer un document WYSIWYG par un langage de commande...

Interaction iconique

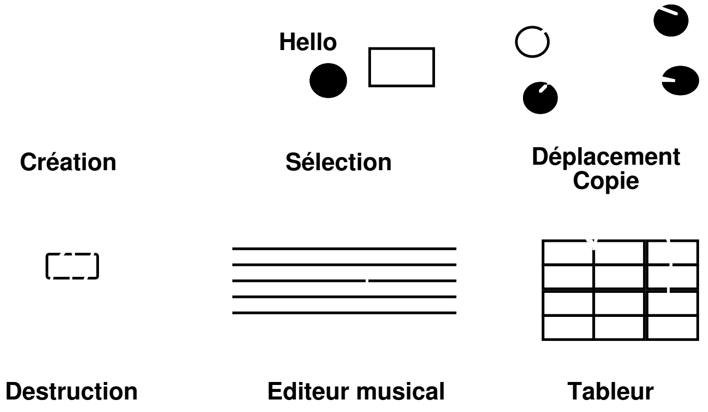
Un icone est une représentation graphique d'un objet ou d'un concept. Les icones doivent être conçus pour être aisément reconnaissables, mémorisables, et différentiables les uns des autres. La stratégie la plus efficace consiste à créer des familles d'icones, comme par exemple sur le Macintosh : la forme générale de l'icone indique s'il s'agit d'un document ou d'une application, le contenu indique de quel type de document il s'agit (texte, graphique...) :



L'interaction iconique consiste à effectuer des commandes par manipulation directe des icones (sélection, déplacement, éventuellement vers un icone) et les techniques usuelles de manipulation indirecte (menus, boîtes de dialogues).

Interaction par reconnaissance de traces

Ce style d'interaction consiste à reconnaître les mouvements du périphérique de localisation par rapport à un vocabulaire gestuel prédéfini. Un geste définit une commande et éventuellement certains de ses paramètres (taille, portée du geste, point de départ et/ou d'arrivée). La position du geste détermine son objet. L'interaction est donc concise, mais invisible : il est difficile de savoir quel est le vocabulaire.



Exemple : de nombreux PDAs (Newton, Pilot, etc.)

Autres styles

D'autres styles d'interaction existent ou émergent. On peut notamment citer la réalité virtuelle, la réalité augmentée, l'interaction multimodale, le collecticiel.

La réalité virtuelle immerge l'utilisateur dans un monde synthétique : tout ce que l'utilisateur perçoit (vue, ouïe et, idéalement, toucher) est produit par le système et inversement toutes ses actions (actions physiques comme parole) sont interprétées par le système. Issue des travaux des militaires dans le domaine des simulateurs de vols, la réalité virtuelle trouve aujourd'hui des applications dans des domaines très spécialisés : médecine, téléopération en milieu hostile, etc.

La réalité augmentée procède d'une approche opposée à la réalité virtuelle : au lieu d'immerger l'utilisateur dans un monde synthétique, on intègre l'interface du système informatique dans les objets et l'environnement quotidiens. Par exemple, une feuille de papier peut servir d'interface : une caméra ou une tablette graphique permet de capturer ce que l'on écrit dessus, et un projecteur permet d'afficher des informations sur la feuille. La frontière entre les mondes physique et informatique s'efface, rendant l'interface invisible.

L'interaction multimodale cherche à exploiter au mieux les capacités d'action et de perception de l'être humain, en s'intéressant notamment aux combinaisons entre modes d'interaction. Par exemple, en entrée la combinaison du geste et de la parole est plus puissante que chacune des modalités prises indépendamment : je peux dire "mets ça ici" en désignant du doigt le "ça" et le "ici" alors qu'il est difficile d'exprimer la même commande seulement par la voix ou par le geste. En sortie, la combinaison de l'image et du son permet par exemple d'attirer l'attention (grâce au son) et de communiquer une information sous forme graphique.

Le collecticiel est une extension de la notion de système interactif dans laquelle plusieurs utilisateurs interagissent avec le système afin de communiquer entre eux via le système. Un exemple d'application collecticielle est un éditeur partagé qui permet à plusieurs personnes de modifier, simultanément, le même document, et de voir, en temps réel, les modifications effectuées par les autres personnes. L'intérêt croissant pour le collecticiel vient de la constatation que la plupart de nos activités sont des activités de groupe, alors que l'usage de l'ordinateur est essentiellement individuel.