

Capitaliser les processus : Maestro

Ma première contribution, qui offre un cadre structurant à toutes les autres, est un outil de gestion de projet de conception dédié aux premières phases de la conception.

9.1 Principes

Cet outil, nommé Maestro s'adresse aux concepteurs débutants et aux formateurs (aide à la transmission de connaissances aux débutants). L'objectif de cet outil est de faciliter la capitalisation des processus de conception de connaissances du domaine (présentement la conception de systèmes interactifs pour différents métiers) mais aussi des techniques de créativité afin de structurer et de guider le processus de conception.

L'objectif est d'aider les concepteurs débutants à être plus créatifs, c'est-à-dire de soutenir l'exploration large. La créativité repose sur des connaissances du domaine et sur les compétences mentales liées à l'imagination [Amabile 1983]. On part du principe que l'on peut aider les concepteurs débutants à être plus créatifs en leur apportant des connaissances du domaine au travers de l'inspiration, et aider l'exploration en les guidant dans les processus créatifs.

Il existe beaucoup d'outils de gestion de projet pour les phases avales d'un projet. Ces outils permettent de gérer les artéfacts produits, leurs différentes versions, de gérer les requis, les choix, la documentation et les bugs par exemples. Cependant, bien qu'utiles, ces fonctionnalités ne soutiennent pas l'exploration large de l'espace de conception, pour arriver à un résultat de qualité. Ma vision est de produire un outil de gestion des phases préliminaires, celles de la recherche d'idées. Cet outil doit guider le concepteur dans le processus créatif. L'idée générale de cet outil est de fournir un support au processus de conception selon trois leviers :

1. La motivation des concepteurs ;
2. La maîtrise du domaine ;
3. La maîtrise des techniques de créativité.

Prenons un scénario pour comprendre cet outil.

9.2 Maestro en action

Alexandre enseigne la conception à des étudiants. Il choisit de leur faire suivre un processus centré utilisateur qui commencerait par une phase de collecte d'informations, puis de génération d'idées, suivie par une phase de sélection et ainsi de suite. Alexandre valide, pour chacune des phases, un ensemble de techniques comme par exemple : le brainwriting, la purge et les matrices de croisement pour la phase de génération. Il utilise pour cela une interface d'administration de l'outil (Figure 15)

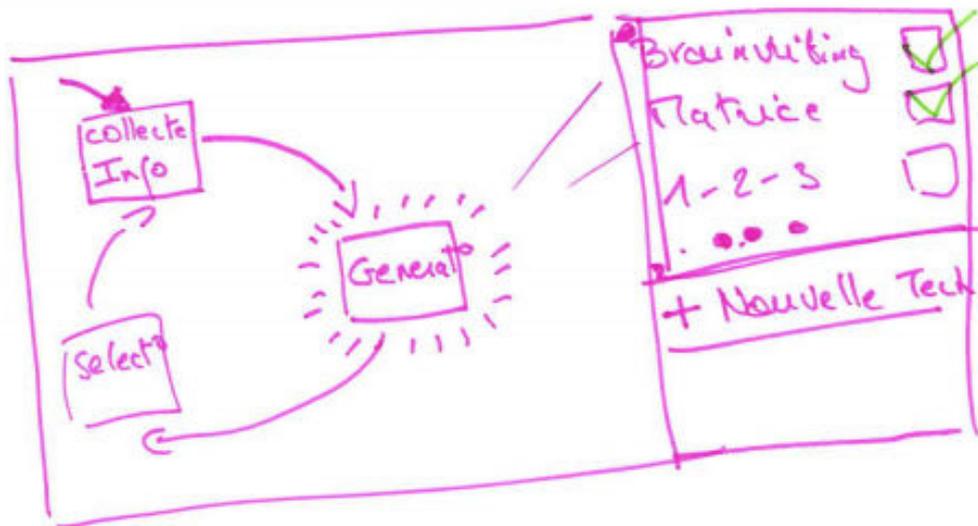


Figure 15. Interface administrateur de Maestro

Les étudiants commencent leur projet. L'outil leur propose plusieurs techniques de collecte d'informations parmi celle validées par Alexandre. Les étudiants choisissent la technique des interviews sur le terrain.

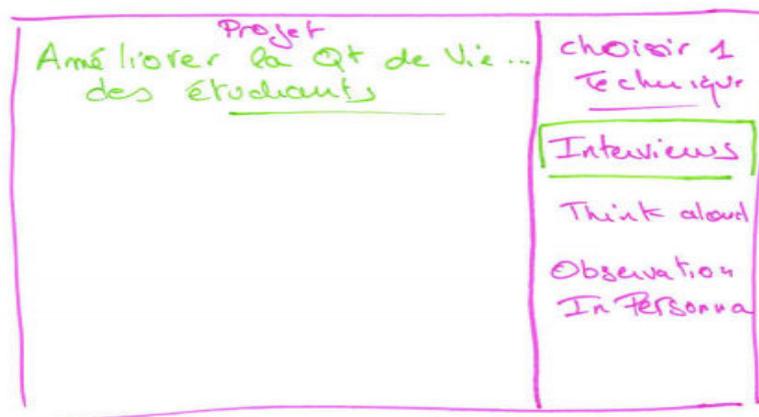


Figure 16. Interface de choix de techniques dans Maestro

L'outil présente le déroulement de l'activité en précisant qu'il est recommandé de faire au moins trois interviews. L'outil leur demande de renseigner des informations sur le contexte (le terrain choisi, les personnes à rencontrer), leur propose une série de questions usuelles pour cette activité et leur propose d'en ajouter. Ces questions seront suggérées aux autres groupes qui s'engageraient dans cette activité. L'outil leur recommande aussi de prendre des photos et d'enregistrer les interviews.

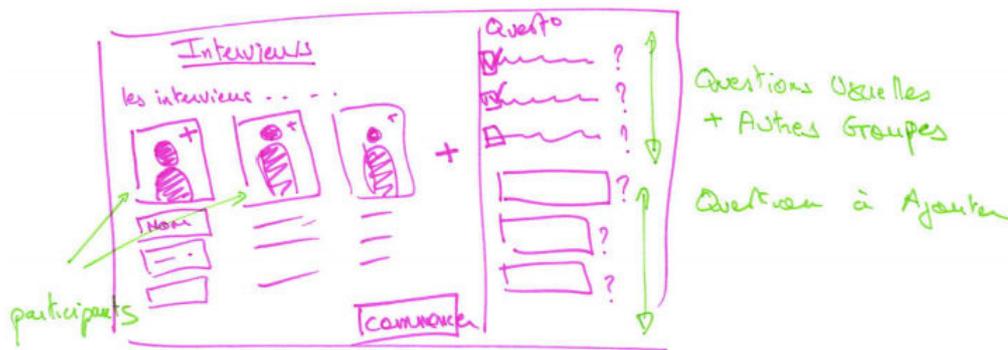


Figure 17. Interface de présentation de la technique "Interviews"

Les étudiants se rendent sur le terrain avec une tablette. Ils enregistrent leurs interviews et ajoutent des fichiers audio et photos directement à l'application depuis la tablette. Une fois de retour en classe, l'application leur propose un débriefing. Les étudiants numérisent leurs notes et transcrivent une partie des fichiers audio qu'ils ajoutent aux artéfacts de l'activité. Ils marquent les artéfacts issus de l'interview numéro 2 comme privés car l'interview était confidentielle. Les autres artéfacts sont accessibles aux autres groupes de la classe qui souhaiteraient s'en inspirer.

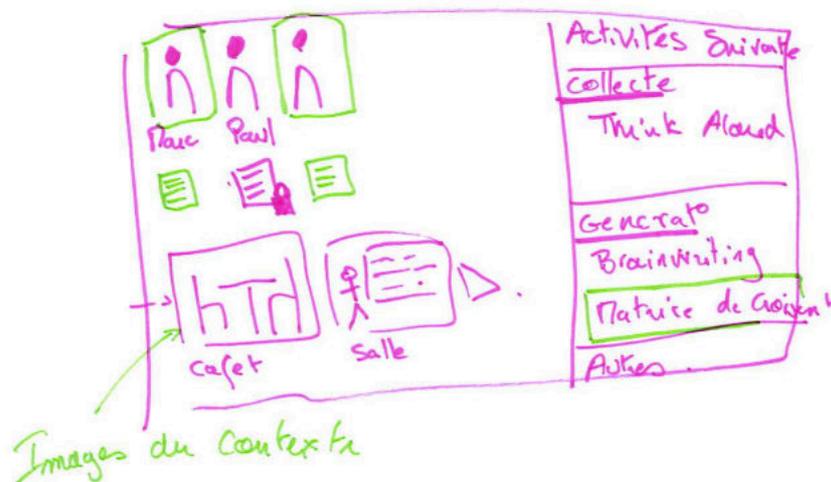


Figure 18. Interface à la fin de la technique "Interviews"

L'outil leur propose de rester dans la phase de collecte d'informations ou de passer à la phase de génération. Les étudiants choisissent la technique de la matrice de croisement : il leur faut choisir deux dimensions. L'outil reprend des artefacts précédents les utilisateurs pour la dimension 1. Il n'y a pour l'instant pas de dimension 2. Les étudiants peuvent en remplir une manuellement ou faire un brainwriting pour trouver des idées de dimensions. S'ils choisissent la deuxième option, l'outil propose d'utiliser un outil externe de brainwriting : par exemple, l'outil de brainwriting avec assistant computationnel présenté dans le Chapitre 11. Les idées produites sont ensuite automatiquement importées dans l'outil.

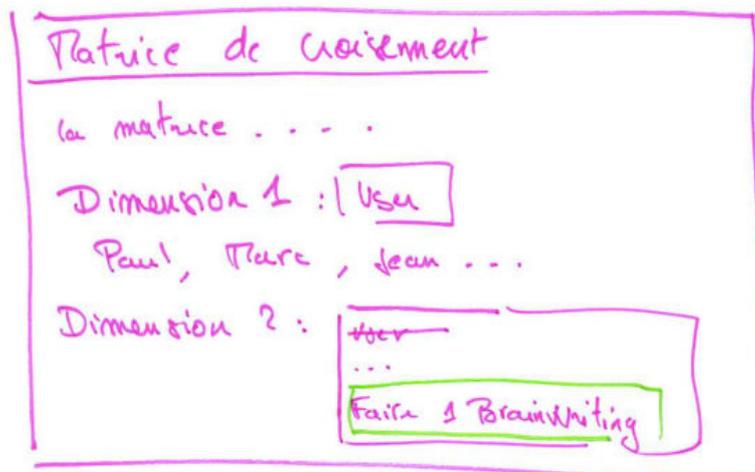


Figure 19. Interface de la technique "Matrice de croisement" dans Maestro

A tout moment du processus, les étudiants peuvent observer les activités réalisées et chercher parmi les artefacts qu'ils ont produits (comme suggéré par [Shneiderman 2007]) mais aussi parmi les productions publiques des autres groupes.

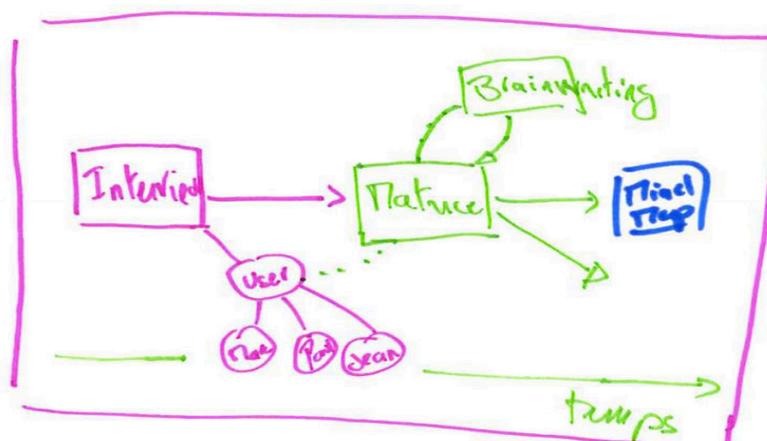


Figure 20. Interface du processus accompli dans Maestro

Les étudiants peuvent reprendre la liste d'idées produites dans l'étape 2 et la *préciser* en la transformant en un modèle des tâches utilisateur en utilisant, par exemple, la notation CTT [Paterno 2003]. Ils peuvent ensuite *concrétiser* ce modèle des tâches en produisant des sketches de l'interface pour finalement abandonner cette piste si elle ne correspond pas aux attentes de l'utilisateur 1 et revenir alors modifier le modèle des tâches pour refléter la nouvelle compréhension du problème. Cet archivage des artefacts produits, ainsi que la conservation des évolutions de ces artefacts et de leurs raisons d'être, appelées Design Rationale, sont des éléments clés d'un outil de support à la créativité. Nous présentons, dans le chapitre 10, un graphe de modèles pour soutenir cette approche.

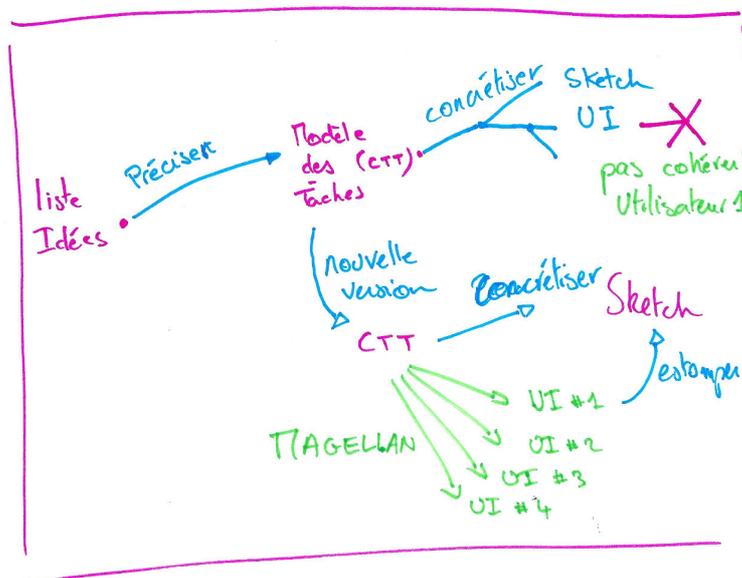


Figure 21. Organisation des artefacts et du Design Rationale dans Maestro

Depuis le nouveau modèle des tâches, les étudiants tentent une exploration automatique des interfaces possibles avec Magellan. Magellan est un outil externe, présenté au chapitre 13, qui génère des variantes d'IHM conformes à un modèle des tâches. Alternativement les étudiants peuvent à tout moment rechercher de l'inspiration dans des galeries d'exemples et artefacts du processus. Nous proposons dans le chapitre 12 une réflexion sur les galeries d'exemples dans le domaine de la conception de sites web et sur les moyens d'automatiser la création, l'organisation et l'exploration d'exemples au sein de galeries basées sur le style d'un site web.

9.3 Discussion

L'outil mis en scène ci-dessus est un outil « idéal » dont seule une version préliminaire a été développée. Elle a fait l'objet du projet de 2^{ème} année Ensimag de Luiza Cicone. Maestro est un prototype web, conçu selon une approche centrée utilisateurs. Les fonctionnalités majeures sont :

- Assurer le guidage dans le processus de conception en donnant accès à des processus et techniques reflétant le savoir-faire du domaine (par exemple, IHM WIMP, sites marchands pour mobile, etc.). L'idée est aussi de se construire une culture commune, avec des références à partager. Les savoir-faire peuvent relever de la communauté des concepteurs d'IHM (par exemple, le WIMP) ou être plus spécifiques à un sous-domaine, à une équipe, etc. ;
- Assurer l'acquisition d'un savoir-faire en matière de techniques de créativité comme le brainstorming, le brainwriting, l'utilisation de galeries, etc. ;
- Enregistrer les artefacts produits tout au long du processus ainsi que les discussions au sein du groupe et pouvoir s'y référer facilement par la suite ;
- Soutenir le travail en collaboration, favoriser le partage des idées et des artefacts. D'une façon générale, l'outil doit favoriser la discussion et le partage de retours d'expérience à tous niveaux. Il sera donc intéressant de mettre en place des forums, chats ou autres outils sociaux dans cette optique.

Les outils décrits dans les chapitres suivants ont tous vocation à être intégrés à Maestro en support aux différentes phases du processus de conception.

The screenshot displays the Maestro web application interface. On the left, a desktop view shows a page titled "Processus centré utilisateur (cours IHM)" with a navigation menu and a main content area. The main content area includes a description of the user-centered design process, a diagram showing the phases (Vision, Analyse, Conception, Evaluation) in a circular flow, and a detailed list of phases with their descriptions. On the right, a mobile app mockup is shown, displaying the same content in a mobile-optimized format. The mobile app has a search bar, a menu icon, and a timeline view of the process phases. The timeline shows the phases occurring over a period from Sun 05 to Sun 11, with a green bar indicating the current phase.

Processus centré utilisateur (cours IHM)

La démarche de conception centrée sur l'utilisateur repose sur l'idée que les utilisateurs finaux sont les mieux placés pour évaluer et utiliser le produit. De ce fait, le développement d'un produit est a priori davantage guidé par les besoins et exigences des utilisateurs finaux, plutôt que par des possibilités techniques ou technologiques. Toutefois, l'utilisateur final peut être entendu de deux manières :

- * l'utilisateur réel, le plus susceptible d'utiliser le produit répondant à ses exigences et étant éventuellement déjà utilisateur d'une version précédente du produit
- * l'utilisateur potentiel qui présente des exigences proches ou équivalentes, et que l'utilisation du produit pourrait intéresser

La définition et le recueil des besoins, des attentes et des exigences applicables au produit doivent être issus d'une démarche rigoureuse dans le cadre d'une intervention ergonomique, d'une enquête utilisateur, d'un test utilisateur. Ces étapes peuvent être effectuées avec un produit existant ou un prototype.

Tags : utilisateur conception

Phases

Phase	Description
Vision	La vision est l'idée de départ du projet.
Analyse	Cette étape vise à préciser les attentes et les besoins des utilisateurs finaux. Elle permet de prendre connaissance de la tâche réelle des utilisateurs et d'analyser le contexte dans lequel ils effectuent, ou vont effectuer, cette tâche. La phase d'analyse permet de préciser l'utilité recherchée par les utilisateurs de l'application. Les méthodes d'observation peuvent comprendre l'utilisation de questionnaires, de grilles d'observation heuristiques, de focus group. Il est préférable d'ordonner ces besoins et exigences selon leur importance vis-à-vis du produit à développer, et des éventuelles contraintes de faisabilité technique
Conception	Sur la base des éléments recueillis dans la phase d'analyse, une première maquette ou prototype du produit est conçu. Cette maquette résulte, d'une part de l'analyse de la tâche des utilisateurs et des spécificités du contexte de travail, et d'autre part des principes et recommandations ergonomiques. Elle peut reposer également sur des standards de conception. Cette première maquette évoluera ensuite en fonction des retours de la phase suivante d'évaluation. Chaque itération permet d'enrichir et finaliser la maquette.
Evaluation	La phase d'évaluation consiste à mesurer l'utilisabilité du produit, autrement dit de valider la satisfaction des utilisateurs dans la réalisation des tâches évaluées. Parmi les différentes méthodes d'évaluation possibles, la principale est le test utilisateur. Celui-ci consiste à placer l'utilisateur en situation d'utilisation réelle du produit et à observer les difficultés rencontrées. L'évaluation permet d'identifier les points à améliorer sur la maquette et donc de préparer la version suivante qui sera à nouveau testée et ainsi de suite. L'expérience montre que 2 à 3 itérations suffisent en général pour finaliser la conception de l'interface.

Chapitre 10. Capitaliser la connaissance en IHM : BANK

Les premières phases de conception nécessitent l'exploration de nombreuses solutions et la proposition de plusieurs variantes de l'interface à concevoir [Buxton 2007]. Un moyen de mener cette exploration est de produire des sketches et des prototypes qui permettent aux concepteurs de rapidement matérialiser leurs idées pour alimenter la discussion, sélectionner et valider des solutions. Ces premières phases sont cruciales : elles favorisent la production d'un « bon » design de base qui va ensuite pouvoir être amélioré et concrétisé. C'est ce qu'exprime [Buxton 2007] lorsqu'il parle de « *Getting the right design before getting the design right* » : il est important de passer du temps à chercher le bon design en restant à un niveau relativement flou de description (sketch, prototype basse fidélité) plutôt que de prendre le premier design et de l'améliorer en profondeur.

Il existe de nombreux outils de sketching (par exemple SILK [Landay 1996], DENIM [Lin 2000], SketchiXML [Coyette 2005] ou Magglite [Huot 2004]) et de prototypage mais, à notre connaissance, il n'en existe pas permettant de capitaliser les artefacts produits et les discussions ou raisonnements suivis, en tout cas à ce niveau du processus de conception. Les travaux visant à capitaliser les connaissances en conception d'IHM s'intéressent aux processus en tant que tels (par exemple, les méthodes centrées utilisateurs), aux « design patterns » (solutions systématiques à des problèmes récurrents), aux boîtes à outils (les widgets sont une forme de capitalisation du savoir-faire pour réaliser des tâches données) et à l'Ingénierie Dirigée par les Modèles (IDM) qui est principalement orientée vers la transformation de modèles d'IHM exprimés à différents niveaux d'abstraction (concepts et tâches, interfaces abstraites, concrètes et finales [Calvary 2004]).

La capitalisation des savoir-faire en IHM et leur utilisation sont un élément important en conception d'IHM mais aussi pour la conception créative (rappelons que la créativité est favorisée par la maîtrise du domaine [Amabile 1983]). Les approches actuelles ne capitalisent que des « produits finis » : comment représenter précisément un bouton ou une barre de navigation. Nous pensons qu'il serait utile de capitaliser aussi des artefacts d'un niveau de précision moindre (par exemple, un sketch décrivant à gros grain la mise en page d'un site web). En effet, un niveau de détail plus faible aide à se concentrer sur l'essentiel et à ne pas se perdre dans les détails [Virzi 1996, Tohidi 2006]. Un moindre degré de précision laisse aussi naturellement plus de place à l'interprétation et à l'improvisation de la part du concepteur mais cela

le guide malgré tout dans une certaine direction. Il nous paraît donc important de proposer un moyen de stocker et d'organiser les sketches et prototypes au même titre que les artefacts plus concrets et précis. Nous nous sommes orientés vers une structure de graphe pour capitaliser ces connaissances, c'est à dire archiver de manière structurée pour pouvoir partager et exploiter de manière manuelle ou automatique ces connaissances dans la suite du processus de conception ou dans d'autres projets de conception.

Le graphe de modèles, nommé BANK pour Bank of Any New Knowledge, est une amélioration du graphe sémantique de [Demeure 2006] pour stocker et réutiliser des composants d'interfaces pendant la conception et l'exécution d'une IHM. Mon objectif est de pouvoir y stocker les artefacts produits au cours du processus de conception, de les lier entre eux de façon pertinente et enfin représenter le savoir-faire dans le domaine.

10.1 Principes

Nous proposons de stocker les savoir-faire et les artefacts produits au cours des processus de conception au sein d'un graphe.

10.1.a Nœuds du graphe

Les nœuds du graphe sont des modèles d'interfaces ou de composants d'interfaces définis à l'un des niveaux d'abstraction du cadre de référence CAMELEON : modèle des Tâches et des Concepts (C&T), Interface Utilisateur Abstraite (AUI), Interface Utilisateur Concrète (CUI) et Interface Utilisateur Finale (FUI). Chaque nœud possède un niveau de précision allant du « sketch premier » à la « définition formelle », ce qui couvre tous les niveaux de fidélité des artefacts des premières phases de conception.

La Figure 23 et la Figure 22 sont des exemples d'artefacts à stocker dans la banque. La Figure 24 montre le positionnement de ces nœuds sur les deux dimensions que sont l'abstraction et la précision. Le nœud M est caractérisé (Figure 23) comme étant une description dans la notation CTT [Paternò 2003] de l'entrelacement de quatre tâches A, B, C, D : c'est donc une description formelle d'un modèle des tâches et concepts. Le nœud N est caractérisé (Figure 22) comme étant le sketch d'une interface concrète correspondant à l'entrelacement des quatre tâches A, B, C, D.

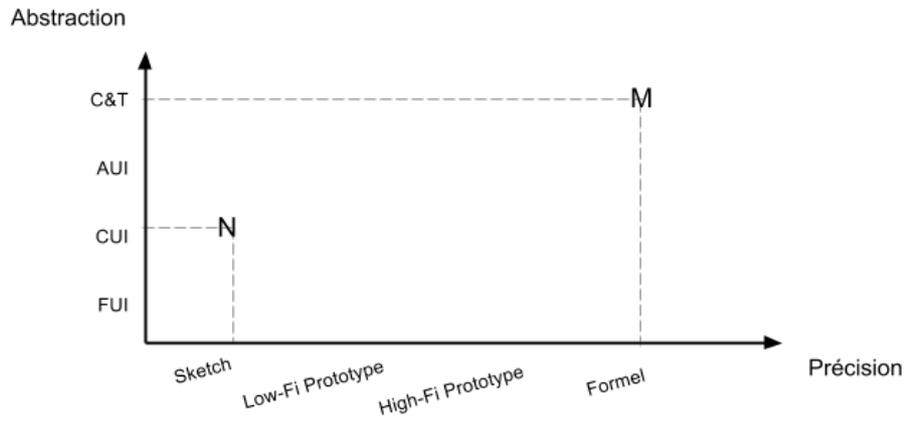


Figure 24. Les noeuds sont caractérisés par le niveau d'abstraction et de précision . M et N sont deux exemples détaillés ci-dessous.

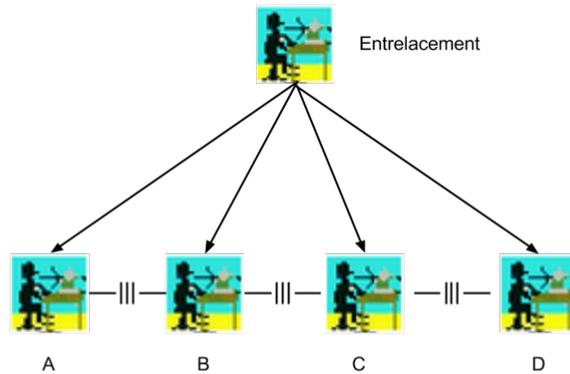


Figure 23. Point M dans la figure 22. C'est un modèle formel d'un entrelacement défini au niveau C&T.

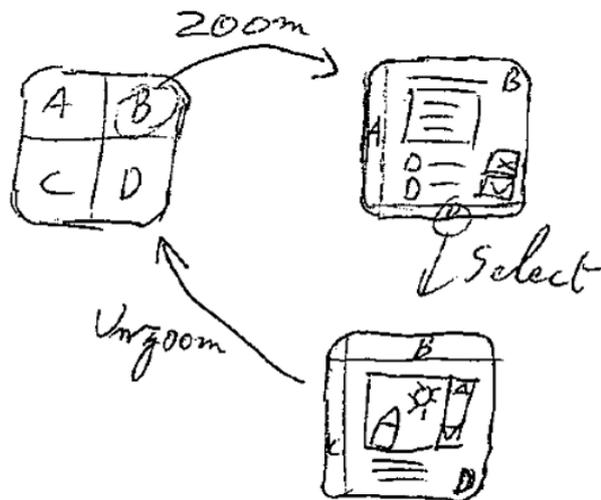


Figure 22. Point N dans la Figure 22: le noeud est un sketch d'entrelacement défini au niveau CUI

10.1.b Arcs du graphe

Les arcs du graphe modélisent les relations entre les modèles d'interfaces. Les arcs peuvent être vus comme les transformations qui produisent le modèle destination à partir du modèle source. Une transformation est définie par :

- Un niveau de précision allant de informel à formel ;
- Le contexte d'usage en termes de plateforme, utilisateurs et environnement que la transformation requiert ;
- Un degré d'originalité qui dénote à quel point le savoir-faire exprimé par l'arc est partagé par les concepteurs. Ce degré va de personnel à global en passant par propre à l'organisation ;
- Un design rationale, c'est-à-dire une justification expliquant les raisons d'être de la transformation.

La Figure 25 illustre la classification des transformations. Les transformations usuelles présentes dans CAMELEON (concrétisation, abstraction) [Calvary 2002] et dans le GDD de [Demeure 2006] sont limitées au niveau d'abstraction (concrétisation et abstraction). Ici, les transformations peuvent aussi être utilisées pour :

- changer le niveau de précision d'un modèle. Soit en *précisant* (sharpen), par exemple en fournissant un modèle formel d'un prototype ; soit, au contraire, en floutant (blur), par exemple, en produisant des sketches pour un modèle formel ;
- Expliciter la composition d'un modèle, par exemple, un modèle des tâches composé de sous tâches ;
- Exprimer qu'un modèle est une variante d'un autre.

Ces transformations sont un moyen d'exprimer le *design rationale* de l'évolution de

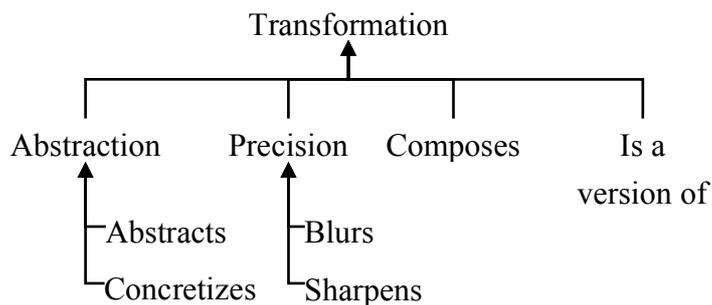


Figure 25: Classification des transformations.

l'interface au cours du processus de conception.

10.2 Bank en action

Cette section développe le potentiel du graphe pour l'exploration de l'espace de conception.

10.2.a A la conception

A la conception, le graphe sert deux buts : 1) inspirer le concepteur en lui fournissant le savoir faire en conception d'interfaces homme machine ; 2) fournir un espace pour stocker et accéder aux interfaces produites pendant la conception. Le graphe produit un moyen pour les équipes de conception de structurer la production de leurs sketches et prototypes. Les relations entre les modèles intègrent le design rationale du processus de conception. La Figure 26 illustre une instance de ce graphe pour un projet de conception d'un système de messagerie instantanée. Les nœuds sont représentés par des carrés, les transformations par des arcs avec une étiquette ovale.

Le projet commence par une réunion entre les concepteurs et le client. Les concepteurs présentent un ensemble d'exemples (A) pour stimuler les clients. A la fin de la réunion, les concepteurs ajoutent un sketch du modèle des tâches et des concepts (B). Ni les tâches ni les concepts ne sont encore bien définis à ce moment, mais les concepteurs et les autres intervenants du projet sont d'accord sur une description informelle du projet. Plus tard, les concepteurs *précisent* cette description en produisant un modèle formel des tâches, sous la forme d'un modèle CTT.

Les concepteurs continuent la conception en explorant l'interface concrète. Les noeuds C, D, E, F et G décrivent les évolutions de la conception. En partant du modèle formel des tâches, les concepteurs explorent deux chemins : C suivi de E puis G, en parallèle avec le chemin F. L'arc est à la fois *concretize* et *blur* : en effet, les concepteurs explorent un niveau plus concret de l'interface (du modèle des tâches à l'interface concrète) au travers d'artéfacts moins précis (d'une description formelle au sketch). Plusieurs versions sont proposées et expliquées. La dernière version (G) précise certaines parties de l'interface.

Le graphe stocke les évolutions, les discussions et les choix au cours du processus. Les concepteurs pourront ainsi, plus tard dans le processus, revenir à l'origine d'une idée ou commencer une nouvelle branche en gardant en mémoire les alternatives. En effet, différentes parties de l'interface peuvent évoluer à différentes vitesses (par exemple dans la version (G)).

Les concepteurs segmentent l'arbre des tâches en trois sous-tâches pour séparer les problèmes pendant la conception : Gérer sa liste de contacts, Gérer son profil et Gérer ses conversations. Ils créent trois nœuds représentant les trois sous-modèles de tâches et les lient avec un arc *composes* au modèle de la messagerie. Ils lient aussi la définition formelle de l'entrelacement de CTT pour préciser comment la composition doit être réalisée. La relation *composes* rend possible la séparation d'un problème en sous problème. Cela est utile pour réduire la complexité et capitaliser des solutions partielles réutilisables dans d'autres projets. Par exemple, il est peu probable que le graphe contienne des nœuds issus des projets précédents pour concrétiser une messagerie instantanée. En revanche, la gestion d'une tâche simple comme l'entrelacement ou la gestion d'une liste peut avoir été abordée, par ailleurs, par la communauté avec la proposition de composants graphiques particuliers tels une Jlist pour une FUI en java ou la balise . Cela permet aussi d'avoir un point d'entrée vers d'autres problèmes liés, par exemple, la recherche dans une liste ou l'ajout et la suppression d'éléments. Il est ainsi possible d'explorer de nouvelles idées.

Les concepteurs peuvent explorer les solutions possibles en composant les solutions aux sous-problèmes. Les sous-problèmes étant récursivement décomposables, l'approche engendre une explosion combinatoire qui milite pour des outils support. Le chapitre 13 présente un outil pour produire des variantes d'IHM. Il s'appuie sur des algorithmes génétiques et la banque de modèles.

Les concepteurs peuvent sélectionner une partie d'un modèle et la lier à d'autres nœuds ou à d'autres parties de modèles. Par exemple, les concepteurs peuvent désigner une partie d'un modèle des tâches comme représentant la tâche "Gérer sa liste de contacts" et lier cette partie au nœud correspondant. Ils peuvent aussi lier cette partie avec sa représentation dans le sketch du nœud C. Cette possibilité d'identifier des parties de modèles est particulièrement utile lorsqu'appliquée avec la transformation *composes*. Les concepteurs peuvent ainsi spécifier qu'un nœud est composé de plusieurs sous-nœuds. Dans le cas d'un modèle des tâches, un sous-nœud peut représenter des sous-tâches impliquées dans le modèle.

10.2.b A l'exécution

A l'instar des concepteurs humains, des algorithmes peuvent s'appuyer sur le graphe pour générer automatiquement des interfaces utilisateur adaptées à un contexte d'usage donné. Il est en effet possible, pour une tâche donnée, de parcourir le graphe pour collecter toutes les implémentations de cette tâche. Pour chaque implémentation, le chemin qui la relie à la tâche d'origine fournit le contexte pour lequel elle a été conçue. Par exemple, dans la figure 4, en suivant les arcs de concrétisation du nœud d'entrelacement, on collecte toutes les solutions existantes pour le représenter. Ce processus peut être guidé par des contraintes émanant, par exemple, du contexte d'usage. Il est ainsi possible de récupérer automatiquement toutes les interfaces concrètes et finales adaptées à un contexte d'usage donné. Cela a été exploré dans [Masson 2010].

Ainsi, BANK peut être utilisé comme fournisseur de composants graphiques et pourrait être intégré dans des algorithmes de génération automatique d'interfaces comme SUPPLE [Gajos 2004]. Plus le graphe est connecté pour une tâche donnée, plus les chances de produire des interfaces adaptées sont grandes. Ainsi, la flexibilité et l'extensibilité de BANK sont des propriétés clefs. Elles sont démarquantes par rapport aux approches explicites qui énumèrent tous les rendus possibles pour des tâches ou des opérateurs de tâches donnés. En fait, les algorithmes comme SUPPLE [Gajos 2004] peuvent être vus comme un arc de concrétisation dans le graphe qui transforme un modèle C&T (avec un niveau de précision formelle), un modèle formel de l'utilisateur (traces, préférences, etc.) et un modèle de plateforme cible (taille de l'écran, composants graphiques utilisables) en une CUI / FUI (avec un niveau de précision formelle). L'application de SUPPLE sur un modèle des tâches particulier correspond à l'ajout d'un arc dans le graphe à partir du nœud qui intègre la description C&T à un nœud qui décrit le CUI/FUI généré. Par exemple, dans la Figure 26, SUPPLE peut être appliqué au nœud C&T qui décrit la messagerie instantanée pour produire un CUI (B dans la Figure 26) optimisé pour la plate-forme P et l'utilisateur U.

10.3 Discussion

BANK permet la capitalisation de tous les artefacts du processus de conception. En allant au-delà de la capitalisation des modèles formels, il dépasse les approches actuelles : il permet d'archiver le processus de conception dans son ensemble et de capturer la raison d'être de la conception.

Cette capitalisation peut servir à la transformation automatique de modèles. Cela a été exploré pour les modèles formels [Gajos 2004, Masson 2010]. Il serait intéressant de transposer l'approche à des modèles moins formels : par exemple, transformer des sketches en interfaces concrètes comme dans SketchiXML [Coyette 2004], mais aussi estomper automatiquement un modèle formel pour effacer des détails et ainsi permettre la dialectique du sketch [Goldschmidt 1991] et un focus sur les aspects importants de l'interface.

Chapitre 11. Suggérer des idées : BrianStorming

Le brainstorming est une des techniques de créativité les plus connues. Bien que la communauté scientifique en ait fait de nombreuses critiques, elle reste largement utilisée dans le domaine de l'IHM. Pourtant, plusieurs expérimentations [Taylor 1958] ont montré que les brainstormings en face à face produisaient moins d'idées que la somme des idées produites individuellement par les participants. [Diehl 1987] rapportent 22 expérimentations dont 18 montrent des résultats similaires. Les 4 restantes n'impliquant que des groupes de 2 ne montrent aucune différence entre la production individuelle et en groupe.

De nombreuses variantes ont été développées pour améliorer le brainstorming. Le brainwriting est une variante du brainstorming où les idées sont écrites par le participant qui les a pensées avant d'être partagées aux autres participants. Les modalités d'écriture et de partage donnent naissance à de nombreuses sous-techniques de brainwriting. Par exemple, dans le brainpooling les idées sont inscrites sur des post-it qui sont directement placés dans une zone commune pour inspirer les autres participants ; dans le 6-3-5 brainwriting, chacun des 6 participants dispose d'une feuille où il inscrit 3 idées en 5 minutes avant de faire passer sa feuille à son voisin de droite, jusqu'à ce que chaque feuille soit passée entre chaque main. On parle de brainstorming électronique lorsque les idées sont enregistrées et partagées au travers d'un système informatique.

De nombreux programmes informatiques ont été développés pour proposer d'améliorer le brainstorming [Petrovic 1994, Stenmark 2001, Faste 2013]. Nous appelons Agent Computationnel d'Assistance au Brainstorming (CABA) un programme informatique capable de produire des "entrées", c'est-à-dire des idées, dans un brainstorming électronique. Nous présentons ici l'outil BrianStorming, un CABA qui s'inspire des idées des autres participants pour en proposer de nouvelles en utilisant le langage naturel. Nous nous intéressons aux questions de recherche suivantes :

1. De nombreux travaux ont montré que l'exposition à des exemples [Smith 1993, Marsh 1996] ou à des suggestions [Bonnardel 1999] permettait d'augmenter le nombre d'idées générées dans une activité créative. Mais dans ces travaux, les suggestions étaient produites par des humains. Ainsi nous nous posons la question de **savoir si un CABA peut augmenter la**

production créative humaine d'un brainstorming électronique en participant à un brainstorming ?

2. [Leggett Dugosh 2005] ont étudié l'influence de l'impact social de suggestions dans un brainstorming. Les participants étaient soumis à l'influence d'idées produites lors de sessions précédentes avant de s'engager dans un brainstorming. Les variables étaient le nombre de suggestions auxquelles les participants étaient soumis, l'originalité des suggestions, ainsi que le fait que les participants étaient informés que les suggestions affichées avaient été soit 1) choisies par un programme informatique parmi l'ensemble des suggestions, soit 2) proposées par un humain ayant un score de créativité proche du participant. L'expérimentation confirme des travaux précédents sur l'augmentation du nombre d'idées générées après exposition à des suggestions mais l'influence de la source des suggestions semble varier selon les conditions. C'est pour cela que, **dans un deuxième temps, nous nous intéressons à l'impact de la connaissance de l'origine des suggestions (CABA ou humaine) dans le brainstorming.** Nous envisageons trois cas : 1) un CABA infiltré (les participants ne sont pas au courant de la présence d'un assistant) ; 2) un CABA anonyme (les participants connaissent sa présence mais ses suggestions sont indiscernables de celles des participants humains), et 3) un CABA explicite (ses suggestions sont clairement identifiées).

Le développement de BrianStorming fut itératif. La première étape a impliqué deux étudiants en 2^{ème} année de l'Ensimag. En quatre semaines, ils ont exploré un prototype d'outil de brainwriting reprenant la métaphore du post-it (Figure 27). Cet outil proposait des suggestions en combinant aléatoirement deux mots issus des propositions des participants. Dans une deuxième étape, j'ai encadré un projet d'Initiation à la Recherche en Laboratoire, réalisé par Alexis Jacob en 2^{ème} année de l'Ensimag. Il a alors expérimenté wordnet. J'ai ensuite terminé la version actuelle de BrianStorming en deux itérations.

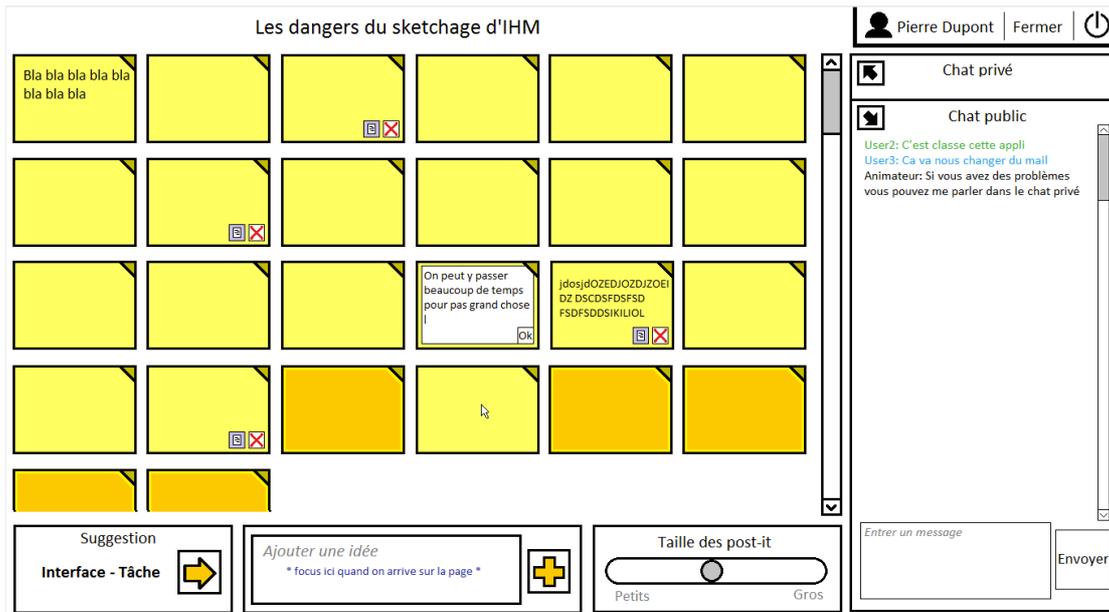


Figure 27. Prototype de brainstorming assisté. Les suggestions sont dans le coin en bas à droite.

11.1 Principes

BrianStorming repose sur un système classique de type clients-serveur web (figure 28). Un serveur de brainstorming gère des sessions de brainstorming. On peut créer une nouvelle session en lui donnant un titre et une durée. Des clients peuvent ensuite s'abonner et poster des messages sur la session. Quand le serveur reçoit un message, il l'ajoute au journal de la session de brainstorming et retransmet le message à tous les participants abonnés à la session.

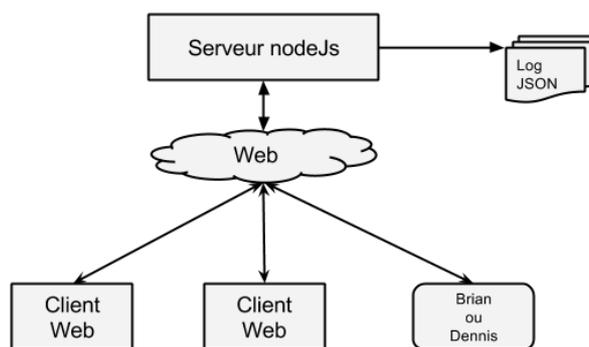


Figure 28. Architecture de BrianStorming

Le serveur et les clients échangent des messages grâce à la bibliothèque javascript socket.io qui permet une communication bidirectionnelle basée sur un système d'événements. Un message est composé d'une phrase, de l'identifiant du client, d'un marqueur temporel, et d'un message de référence. Le message de référence reprend le concept de fil d'idées emprunté à [Jaco 2013] : on rebondit sur une idée précédente en liant l'idée courante à celle-là.

Deux types de clients sont actuellement implémentés. Le premier type correspond aux CABA ; le second à des pages web permettant aux utilisateurs de participer aux sessions. Pour ces derniers, j'ai exploré deux visualisations de sessions. La première dispose d'une zone de texte pour rentrer ses idées et une zone d'affichage de toutes les idées proposées par tous les participants au cours de la session. Cette zone d'idées est composée de plusieurs colonnes pour maximiser le nombre d'idées affichables à un instant donné. Les idées des autres participants sont anonymes, c'est-à-dire qu'on ne sait pas qui les a proposées, ni même si deux idées données sont issues du même participant. Seules les idées proposées par l'utilisateur du client apparaissent de façon distinctes (Figure 29). La seconde visualisation reprend le principe du fil d'idées [Jaco 2013] et explore une représentation en graphe qui indique les relations existantes entre idées. L'utilisateur peut ajouter une idée à une autre en cliquant dessus (Figure 30).

Les autres clients sont des agents informatiques. Ces agents peuvent s'abonner et poster des idées comme les participants humains. Il existe, pour l'instant, deux agents informatiques Brian et Dennis⁷. Brian s'appuie sur les relations entre idées et entre mots utilisés pour proposer ses suggestions. Dennis utilise un mécanisme beaucoup plus simple. Dennis et Brian sont codés en javascript et exécutés sous l'environnement NodeJS (nodejs.org).

⁷ Brian est un anagramme de Brain et provient de constantes erreurs de frappe à l'exécution dues au fait que ma main droite tape plus rapidement que ma main gauche. Brian est aussi un personnage de manuel scolaire pour apprendre l'anglais, son ami dans l'histoire s'appelle Dennis. Ni l'un, ni l'autre ne sont très intelligents dans mon souvenir...



Figure 29. Interface du client simple.

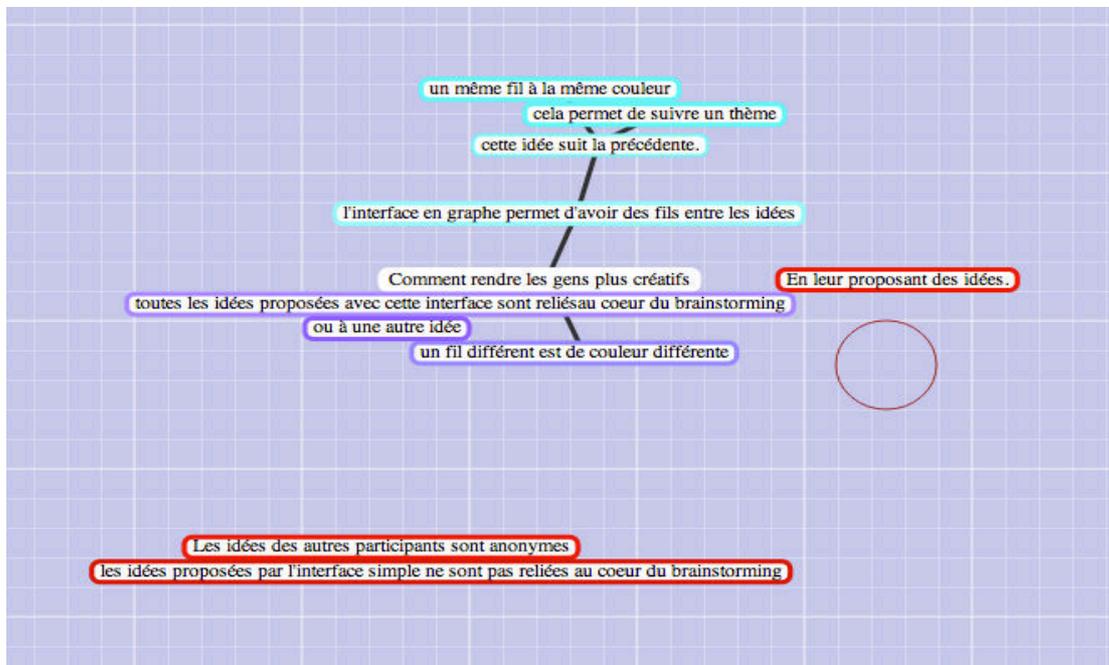


Figure 30. Interface du client sous forme de graphe

11.2 BrianStorming en action

Dennis et Brian commencent par s'abonner au serveur et reçoivent les phrases proposées par les autres clients. Ces phrases sont ensuite envoyées à un serveur de Google translate pour être traduites en anglais. Les phrases anglaises obtenues sont alors segmentées et étiquetées selon leur classe grammaticale, et lemmatisées par deux bibliothèques javascript pos-js⁸ et natural⁹. On obtient alors une liste de lemmes et leurs rôles dans la phrase d'origine. A partir de là, le comportement de Brian et celui de Dennis diffèrent.

11.2.a Dennis

Dennis est la version simple d'un agent d'assistance au brainstorming. L'idée était de tester le concept d'assistant au brainstorming qui proposerait des suggestions en cours de brainstorming. C'est aussi un prototype pour tester la chaîne complète du processus (brainstorming, traduction, proposition) et de tester les différentes modalités de présentation des suggestions aux utilisateurs. Il cherche à proposer des phrases basées sur une structure préfabriquée avec de nouveaux mots reliés aux mots trouvés dans les phrases proposées par les autres clients.

Pour fabriquer des suggestions grammaticalement correctes, Dennis utilise une liste de phrases préfabriquées appelées *structures* (Figure 31) dont il extrait la structure des phrases qu'il doit construire. Comme pour les phrases proposées, Dennis identifie les noms, verbes, adjectifs et adverbes et les marques « à remplacer ». Il garde les déterminants, les prépositions et autres éléments structurants de la phrase. Cette approche permet une meilleure lisibilité des structures utilisées par Dennis. Elle accélère la configuration de Dennis puisqu'il est plus rapide de conceptualiser « something as strong as a bull » plutôt que « <noun> as <adjective> as a <noun> » et, en conséquence, d'ajouter une phrase de ce type.

⁸ <https://github.com/dariusk/pos-js>

⁹ <https://github.com/NaturalNode/natural>

Dennis construit au fur et à mesure du brainstorming quatre listes : des noms, des verbes, des adjectifs et des adverbes. Il construit ces listes à l'aide de Wordnet. Wordnet est une ressource lexicale sémantique construite comme un graphe de mots reliés par des relations. Les relations principales sont l'hyponymie et l'hyperonymie. L'hyperonymie est la relation hiérarchique, ce qui signifie qu'un mot est plus général qu'un autre. Par exemple, un bateau est un hyperonyme de voilier. Inversement, un voilier est un hyponyme de bateau. On trouve aussi dans Wordnet les relations suivantes : antonymie, synonymie, holonymie et meronymie (une partie d'un ensemble, bateau est un holonyme de voile, et voile un meronyme de bateau) ainsi qu'un certain nombre d'autres relations mineures. Dennis remplit ses listes en parcourant aléatoirement le graphe en prenant comme point de départ chaque nouveau mot découvert dans les phrases des autres clients. Dennis limite son exploration du graphe à un maximum de 16 liens traversés par rapport au mot de départ.

Avant d'ajouter le mot à sa liste, Dennis vérifie qu'il existe une traduction connue par Google pour ne pas ajouter des mots qui provoqueraient des phrases intraduisibles. En effet, Wordnet connaît des mots particulièrement étranges. Par exemple, *futureness* est un mot qui connote la qualité d'exister dans le futur et dont il n'existe pas de traduction française exacte, ou le cas échéant cette traduction n'est pas connue de l'outil de traduction de Google. Google est un traducteur statistique, capable de traduire si les phrases sont apparues souvent dans son corpus. Pour ne pas avoir que des mots liés à ceux déjà proposés, surtout au départ lorsque peu ont été proposés, Dennis commence avec une base tirée des 1000 mots les plus utilisés dans la langue anglaise, épurée des articles, prépositions, etc. (408 noms, 218 verbes, 111 adjectifs, et 100 adverbes présentés en annexe).

- "a strong bull eats"
- "the strong bulls eat"
- "a bull that can eat something"
- "the bulls that can eat something"
- "a bull eating strongly"
- "the bulls who are eating strongly"
- "something as strong as a bull"
- "something as strong as bulls"
- "a bull with an apple"
- "something with an apple"
- "an apple with something"
- "bulls with an apple"
- "a bull with apples"
- "bulls with apples"

Figure 31. Liste des phrases-structures de Dennis

Dennis possède donc une liste de phrases-structures et une liste de mots pour les remplir. Dennis choisit aléatoirement une phrase et remplit les trous avant de soumettre sa phrase au serveur. Dennis essaye de parler en moyenne autant que les autres participants. La Figure 32 illustre un exemple de session à laquelle a participé Dennis. Les phrases en vertes sont les propositions du participant qui a fait la capture d'écran, les phrases en noires sont les propositions des autres participants du brainstorming et finalement en rouge les suggestion de Dennis.

11.2.b Brian

Plutôt que de grouper les noms, verbes, adjectifs et adverbes de façon non structurée, Brian maintient les structures déjà présentes dans les idées produites par les participants au cours du brainstorming. Par exemple, Brian va essayer de combiner des mots qui apparaissent dans une même phrase, ou qui proviennent de la même personne, ou du même fil d'idées.

Le fonctionnement de Brian est fortement inspiré de Copycat [Vanschoren 2004], un modèle informatique du raisonnement par analogie. Comme Copycat, Brian possède trois structures de données qui représentent la vision du brainstorming, les connaissances et les actions (pensées) de Brian. Ces structures sont :

- Un *espace de travail*, qui est une représentation structurée sous forme de graphe des participants, des phrases et des mots présents dans le brainstorming. Cet espace contient aussi les phrases qu'il construit en vue de les proposer. L'espace de travail est un graphe où les nœuds sont de type personne, phrase, mot, phrase à trous ou proposition (une phrase construite par Brian mais pas encore soumise au brainstorming). A chaque nœud correspond une valeur d'activation qui représente l'attention que lui porte Brian à un instant donné. Les nœuds sont reliés entre eux par les relations d'appartenance (d'une phrase à un participant, d'un mot à une phrase) et de succession (de phrases, soit temporellement, soit dans un fil d'idées s'il en existe).

Moi : Mes idées sont clairement marquées
les idées des autres sont anonymes
Les suggestions de Brian et Dennis peuvent être anonymes ou pas
Moi : Dans cet exemple Dennis est en mode visible
Suggestion : un nombre qui peut contenir quelque chose
Moi : lapin
Moi : chat
Moi : chien
Moi : vache
Moi : cochon
Moi : veau
Moi : cheval
Suggestion : des chevaux qui peuvent décider quelque chose
Moi :
Moi :
Moi :
Moi :
Suggestion : titres avec attentions
Suggestion : quelque chose avec un verre

Figure 32. Exemple de suggestion de Dennis

- Un *dictionnaire* qui contient les lemmes que connaît Brian ainsi que leurs relations. Ici, un lemme s'entend comme une entité sémantique et correspond à une entrée précise dans le Wordnet. Les lemmes du dictionnaire sont reliés entre eux par les relations trouvées dans Wordnet (synonymie, homonymie etc.). Il existe une relation entre les mots de l'*espace de travail* et les lemmes du *dictionnaire*. Plusieurs mots peuvent correspondre à un même lemme (e.g. boat, Boat et boats peuvent tous les trois correspondre à l'entrée boat%1:06:00::¹⁰), plusieurs lemmes/entrées dans le wordnet peuvent correspondre à un même mot (e.g. boat peut correspondre à boat%1:06:00:: (l'embarcation), boat%1:06:01:: (une saucière) et boat%2:38:00:: (le verbe prendre le bateau)).
- Une *fourmilière* qui contient des fourmis. Ce sont des agents indépendants, seuls capables de modifier l'espace de travail et le dictionnaire. Il existe par exemple une fourmi capable d'ajouter les nouvelles phrases des participants à l'espace de travail, des fourmis capables d'ajouter un nouveau lemme tiré du Wordnet pour un mot de l'espace de travail, des fourmis capables d'explorer le Wordnet pour ajouter des nouveaux lemmes au dictionnaire, des fourmis pour construire des phrases à trous à partir des phrases du workspace ou encore des fourmis pour créer des propositions.

Chaque fourmi est associée à des nœuds dans l'espace de travail et dans le dictionnaire. Le nœud du dictionnaire représente la modification qui va être effectuée sur le nœud de l'espace de travail par la fourmi. Par exemple, une fourmi qui cherche des synonymes dans le Wordnet est associée au nœud synonyme et aux lemmes sur lesquels elle veut opérer. Brian calcule une activation pour chaque fourmi basée sur l'activation des nœuds auxquels elle est associée.

Le processus de Brian est itératif. A chaque itération, une fourmi est choisie au hasard avec une probabilité proportionnelle à son activation parmi celles présentes dans la

¹⁰ L'entrée se lit ainsi lemme%lex_sense. Où lex_sense est composé dans l'ordre

- de la classe grammaticale du mot (1 pour un nom, 2 pour un verbe...)
- du fichier (parmi les 45 fichiers organisés en fonction de la classe grammaticale et de la sémantique) contenant l'entrée (e.g. 06 noun.artifact nouns denoting man-made object, 38 verb.motion verbs of walking, flying, swimming)
- un id à deux chiffres, commençant à 00 de telle sorte que lemma.id soit unique

et deux champs spécifiques aux adjectifs satellites, ici vides.

fourmilière. La fourmi s'exécute et modifie l'espace de travail et/ou le dictionnaire et rajoute potentiellement des fourmis dans la fourmilière pour continuer son travail. A la fin de l'exécution d'une fourmi, l'activation est propagée parmi les nœuds de l'espace de travail et du dictionnaire. L'activation d'un nœud se diffuse dans tous les nœuds auxquels il est relié. L'activation globale de l'espace de travail et du dictionnaire est maintenue à un niveau constant, soit en réduisant l'activation de tous les nœuds si l'activation augmente, soit en activant aléatoirement un nœud (l'attention de Brian se porte sur un nouveau sujet). Finalement l'activation des fourmis est recalculée en fonction de l'activation des nœuds.

11.3 Discussion

Vis-à-vis des deux questions de recherche posées au début du chapitre (un CABA peut-il augmenter la production créative humaine d'un brainstorming électronique en participant à un brainstorming ? Quel est l'impact de la connaissance de l'origine des suggestions - CABA ou humaine - dans le brainstorming ?), notre hypothèse est que l'utilisation d'un CABA améliore la créativité des participants humains, qu'il soit explicite ou implicite. Afin de valider cette hypothèse, nous avons procédé à une expérimentation.

Les participants, des étudiants en informatique de l'Université Joseph Fourier, sont invités par groupes de quatre dans une salle de l'université équipée d'IMAC. Les participants sont placés aux quatre coins de la pièce pour réduire les communications en dehors de l'outil et éviter qu'ils puissent identifier l'origine des idées dans le brainstorming. Les participants sont informés qu'ils vont tester un système de brainstorming électronique. L'examineur rappelle les règles du brainstorming et invite les participants à remplir un questionnaire comprenant nom, âge, sexe et familiarité avec les brainstormings. Les participants commencent ensuite la session par deux échauffements pour se mettre en condition :

1/ l'exercice dit « de la brique » qui consiste à produire le plus d'usages alternatifs pour une brique pendant une minute. Cet exercice est un classique des tests de fluidité ;

2/ Un exercice d'association où une liste de 10 couples de mots est présentée. Pour chaque couple, le participant doit proposer le ou les premiers mots qui lui viennent à l'esprit. Cet exercice est conçu pour inciter les participants à rebondir sur les idées.

Après ces deux échauffements, les participants s'entraînent pendant 5 minutes sur l'outil. Les participants sont invités à poser des questions sur le fonctionnement de l'outil, puis l'examineur leur rappelle les règles et leur présente, si nécessaire, les détails de la condition expérimentale :

Groupe contrôle - Brainstorming électronique de référence : 4 participants utilisent l'interface web pour faire un brainstorming. Aucune suggestion n'est soumise par le CABA.

Condition A - CABA infiltré : idem que la condition contrôle, mais avec 5 participants. Un CABA joue le rôle du cinquième participant et fournit des propositions. Les participants ne sont pas conscients de la présence d'un cinquième participant, ni donc du fait qu'il s'agit d'un CABA.

Condition B - CABA anonyme : idem que la condition A, mais les participants sont prévenus de la présence d'un cinquième participant CABA. Cependant, les propositions du CABA ne sont pas discernables de celles des autres participants.

Condition C - CABA explicite : idem que la condition B mais les propositions faites par le CABA sont clairement discernables (la couleur du texte est modifiée).

Le brainstorming commence alors et dure 30 minutes. Après le brainstorming, les participants sont invités à remplir un rapide questionnaire sur leur ressenti du brainstorming, puis l'examineur révèle la présence du CABA dans la condition A.

Nous avons commencé les expérimentations, mais dû à des problèmes logistiques, nous n'avons pu en conduire que 3 au moment de la rédaction de ce manuscrit. Les résultats ne sont pas au moment de l'écriture de ce manuscrit présentable du à ces problèmes, mais les retours des participants sont encourageant. Une nouvelle campagne d'expérimentation sera lancée courant 2015.