L’ergonomie des logiciels

L’ergonomie des logciels est un sujet très discuté. Et je dois confesser que je ne suis pas un professionnel du sujet (je suis concepteur et développeur, mais pas ergonome).

Aussi, je ne chercherai pas dans le présent article à étaler une science que je n’ai pas. Je voudrais principalement faire valoir que l’on peut considérer l’existence de deux approches différenctes de l’ergonomie :

- **L’ergonomie de la simplicité et de l’intuitivité**, la plus reconnue en ce qu’elle s’adresse au grand public.
- **L’ergonomie de l’efficacité**, moins reconnue en ce qu’elle s’adresse à un public de spécialistes.

1. L’ergonomie de la simplicité et de l’intuitivité

Cette approche est celle à laquelle on pense généralement quand on parle d’ergonomie « tout court » dans le monde du logiciel.

Dans le monde du logiciel propriétaire, c’est notamment celle qui a fait la renommée de la firme Apple. Et dans le monde du logiciel libre, c’est un marqueur revendiqué de l’environnement de bureau Gnome.

Dans cette approche, il s’agit essentiellement de proposer des interfaces **graphiques**, pas trop encombrées pour ne pas noyer l’utilisateur et dotées de moyens de contrôles pensés de sorte que :

- Leurs fonctions soient facilement identifiables, notamment au moyen d’icônes évocatrices.
- Ils soient positionnés aux endroits auxquels on attendrait intuitivement qu’ils le soient.
- Ils soient d’autant plus faciles d’accès que leurs fonctions sont fréquemment utilisées.
- Ils soient le mieux adaptés possibles aux dispositifs matériels par le biais desquels ils vont être utilisés : claviers et souris, écrans tactiles de tailles diverses (téléphones, tablettes…).

  Par exemple un bouton disposé sur un écran tactile de téléphone devra être suffisamment gros pour que l’on puisse l’activer avec un doigt. Ceci pourra l’amener à être plus gros qu’un bouton destiné à être activé à la souris sur un écran d’ordinateur, quand bien même on dispose de moins de place sur un écran de téléphone que sur un écran d’ordinateur.

Il s'agit en fin de compte que le logiciel soit aussi facile à utiliser que possible, **même sans formation spécifique** de l'utilisateur. Cela requiert notamment que les différentes fonctions qu'il propose soient facilement **découvrables**.

Le maître mot serait que l'on veut que **ça coule de source**.

2. L'ergonomie de l'efficacité

L'ergonomie de l'efficacité est une approche radicalement différentes.

Il ne s'agit plus que cela coule de source pour un utilisateur peu ou pas formé. Il s'agit au contraire que le temps que l'utilisateur « perdra » à se former à l'utilisation du logiciel soit un investissement qui s'avérera ultérieurement payant en gains d'efficacité.

Cela passera parfois (et même souvent) par des mécanismes complètement contre-intuitifs, tels que des raccourcis abscons à mémoriser ou des techniques à maîtriser.

Il y a donc un **effort** à fournir, une **courbe d'apprentissage** plus ou moins raide à « se farcir » avant d'être capable d'utiliser correctement le logiciel. Mais une fois qu'on l'est, on est (en principe) bien plus efficace avec ce logiciel qu'on ne le serait avec un logiciel remplissant les mêmes fonctions en ayant adopté une approche ergonomique plus classique.

Un tel effort initial ne vaut le coût d'être fourni que pour des logiciels dont on fera une utilisation suffisament intensive pour que les gains d'efficacité permettent, au bout du compte, de rattraper le temps perdu à se former. Cette approche s'adresse donc plutôt à des spécialistes, concernant des logiciels relatifs à leur(s) spécialité(s).

2.1. Un premier exemple avec Vi(m)

Pour prendre mon exemple personnel, l'une de mes spécialités en matière d'utilisation de l'informatique est de taper du texte. Je le fais dans des langages de programmation en tant que développeur de logiciels. Et je le fais également en français quand je rédige des articles comme celui que vous êtes présentement en train de lire.

En tant qu'adepte convaincu de cette approche que j'appelle ici ergonomie de l'efficacité, j'ai donc décidé d'investir du temps à apprendre à me servir d'un éditeur de texte qui en est un bon représentant : **Vi**, ou plutôt **Vim** qui est sa version améliorée et ses différentes déclinaisons (**gVim**, **neovim**…).

Je suppose que tous les utilisateurs de ce logiciel se souviennent de la brutalité de leur premier contact avec lui. C'est certainement mon cas : la première fois qu'on m'a fait utiliser Vi, quand j'étais étudiant en développement logiciel, ma première pensée a immédiatement été : « mais qu'est-ce que c'est que cette m…, c'est du délire de proposer **ça** !? ».

De prime abord, cet éditeur de texte semble être né d'un esprit malade, à destination d'un public de fous furieux, puisqu'il s'avère qu'il y a pourtant bien des gens qui l'utilisent. Rien ne semble aller : alors qu'on s'attendrait, une fois l'éditeur lancé, à immédiatement pouvoir taper du texte, on apprend que l'on ne peut pas parce qu'on se trouve en « mode normal » et que l'on doit d'abord passer en « mode insertion » au moyen d'un raccourci clavier à retenir. On découvre par la suite qu'il existe une quantité

invraisemblable de ces raccourcis clavier et autres commandes à taper et qu'il faudra même apprendre à les combiner entre elles pour obtenir mille et un effets possibles.

Aujourd'hui, je ne peux plus me passer de cet éditeur de texte et je vis comme une souffrance les moments dans lesquels je dois épisodiquement utilser des éditeurs de texte plus classiques.

Une fois qu'on les maîtrise suffisament, des gestes ou des techniques intellectuelles peuvent devenir une sorte de seconde nature, quand bien même elles pouvaient initialement paraître parfaitement contre-intuitives.

Dans le cas de Vim, après avoir passé le cap de la prise de contact, on s'aperçoit petit à petit que ces différentes commandes, qui finissent par tomber naturellement sous les doigts, sont en fait d'une concision ciselée avec une extrême précision et permettent de réaliser en quelques caractères tapés des opérations d'édition de texte qui, avec un éditeur de texte plus candide, auraient nécessité de nombreuses et fastidieuses manipulations.

Un petit exemple abstrait, supposons que vous vouliez constituer un fichier de texte avec 1 million de fois la ligne : « Vous me le copierez 1 000 000 fois. ».

Voici comment je procéderais avec un éditeur basique :

1. Taper la ligne « Vous me le copierez 1000000 fois. ».
2. Sélectionner la totalité du texte (typiquement avec le raccourci clavier `Ctrl+a`).
3. Copier la ligne ainsi sélectionnée (typiquement avec le raccourci clavier `Ctrl+c`).
4. **A 9 reprises**, coller la lignes (typiquement avec le raccourci clavier `Ctrl+v`). On se retrouve alors avec 10 lignes.
5. Sélectionner **à nouveau** la totalité de ces 10 lignes (`Ctrl+a`).
6. Copier **à nouveau** ces 10 lignes copiées (`Ctrl+c`).
7. **A 9 reprises à nouveau**, coller les 10 lignes (`Ctrl+v`). On se retrouve alors avec 100 lignes.
8. Sélectionner **à nouveau** la totalité de ces 100 lignes (`Ctrl+a`).
9. Copier **à nouveau** ces 100 lignes copiées (`Ctrl+c`).
10. **A 9 reprises à nouveau**, coller les 100 lignes (`Ctrl+v`). On se retrouve alors avec 1000 lignes.
11. Sélectionner **à nouveau** la totalité de ces 1000 lignes (`Ctrl+a`).
12. Copier **à nouveau** ces 1000 lignes copiées (`Ctrl+c`).
13. **A 9 reprises à nouveau**, coller les 1000 lignes (`Ctrl+v`). On se retrouve alors avec 10000 lignes.
14. Sélectionner **à nouveau** la totalité de ces 1000 lignes (`Ctrl+a`).
15. Copier **à nouveau** ces 1000 lignes copiées (`Ctrl+c`).

16. **A 9 reprises à nouveau**, coller les 1000 lignes (`Ctrl+v`). On se retrouve alors avec 10000 lignes.
17. Sélectionner **à nouveau** la totalité de ces 10000 lignes (`Ctrl+a`).
18. Copier **à nouveau** ces 10000 lignes copiées (`Ctrl+c`).
19. **A 9 reprises à nouveau**, coller les 10000 lignes (`Ctrl+v`). On se retrouve alors avec 100000 lignes.
20. Sélectionner **à nouveau** la totalité de ces 100000 lignes (`Ctrl+a`).
21. Copier **à nouveau** ces 100000 lignes copiées (`Ctrl+c`).
22. **A 9 reprises à nouveau**, coller les 100000 lignes (`Ctrl+v`). On se retrouve alors **enfin** avec les 1000000 lignes voulues.

Maintenant, avec Vim :

1. Passer en mode insertion (`i`).
2. Taper la ligne « Vous me le copierez 1000000 fois. ».
3. Repasser en mode normal (`Esc`).
4. Taper la commande `yy999999p`. On se retrouve alors avec les 1000000 lignes voulues.

2.2. Un second exemple avec Xmonad (et autres gestionnaires de fenêtres pavants)

Le grand public utilisateur d'ordinateurs est habitué à des environnements graphiques à base de fenêtres. C'est d'ailleurs de cela que le système d'exploitation propriétaire Windows tire son nom. Mais il en va de même pour son concurrent non moins propriétaire MacOS ou les environnements de bureau les plus répandus du monde Linux que sont Gnome et KDE Plasma.

Dans tous ces environnements graphiques, les fenêtres sont pourvues de bordures et de barres de contrôle qui permettent de les redimensionner et de les déplacer à la souris. Par ailleurs, ces déplacements peuvent amener ces fenêtres à se chevaucher les unes les autres.

Mais il existe, dans le monde Linux et généralement inconnue du grand public, une autre famille de gestionnaires de fenêtres qui est celle des gestionnaires **pavants** (« tiling window manager » en anglais).

Dans ces environnements graphiques, les fenêtres ne sont plus redimensionnées et déplacées à la souris. Au lieu de cela, elles sont automatiquement agencées par le gestionnaire de fenêtres pour se juxtaposer les unes à côté des autres, sans se chevaucher, de sorte à remplir l'écran.

Des raccourcis claviers permettent généralement de changer le principe géométrique de l'agencement des fenêtres. Par exemple, un agencement classique consiste à avoir une fenêtre principale qui occupe la moitié gauche de l'écran et toutes les autres

fenêtres se partageant à parts égales la moitié droite de l'écran, en s'empilant verticalement les unes au dessus des autres. Un raccourci peut alors changer le principe de cette répartition en accordant à la fenêtre principale la moitié supérieure de l'écran au lieu de sa moitié gauche et aux autres fenêtres sa moitié inférieure au lieu de sa moitié droite, qu'elles se partagent alors horizontalement plutôt que verticalement.

Un tel gestionnaire est inutilisable sans retenir un certain nombre de raccourcis clavier. C'est en fait son principe même : le but est précisément d'utiliser plus intensément le clavier pour ne pas avoir à recourrir à la souris. Ainsi, en supposant que les programmes que vous utilisez au sein de ces fenêtres fassent également la part belle aux interactions via le clavier (et c'est typiquement le cas si vous passez le plus clair de votre temps à utiliser un éditeur de texte), vous pourrez justement vous concentrer sur le clavier et ne pas perdre de temps à passer du clavier à la souris, puis au clavier à nouveau.

Ce passage du clavier à la souris, puis au clavier à nouveau, ne paraît pas grand chose. Mais considérant que vous pouvez devoir le faire plusieurs centaines de fois par jour, les secondes perdues s'accumulent pour former des minutes, des heures, puis des jours…

A titre personnel, j'utilise Xmonad, qui appartient à cette famille. Mon choix s'est porté sur celui-ci principalement parce qu'il a été réalisé avec le langage de programmation Haskell et qu'il se configure avec ce même langage, dont je suis féru.

Mais il en existe d'autres, comme Awesome, i3 ou encore dwm.

3. Un principe plus général

Cette dualité entre le simple et immédiat d'une part et le plus compliqué mais plus efficace d'autre part dépasse le seul cadre de l'utilisation de logiciels. On en trouve des exemples dans de nombreux autres domaines de l'activité humaine qui présentent un caractère technique.

C'est par exemple le cas dans le domaine des sports :

- La méthode saut en hauteur inventée par Dick Fosbury et qui lui a permit de remporter le titre olympique en 1968 est tout sauf intuitive. Et avant cela les champions de saut en hauteur utilisaient tous des méthodes qui pouvaient paraître plus naturelles, comme le saut en ciseaux.

  Àujourd'hui, tous les athlètes de cette disciplines s'entraînent à maîtriser cette technique, si efficace qu'il n'existe plus aucune chance de remporter une compétition sans l'employer.

- Le principe même des arts martiaux est d'employer des techniques plus ou moins sophistiquées pour obtenir sur ses adversaires un ascendant que ne permet pas la condition physique seule.

On trouve évidemment bien d'autres exemples dans des domaines divers et variés comme les sciences, les techniques et l'artisanat. Au bout de cette réflexion, on trouve la simple constatation, qui sonne comme une évidence, qu'il existe des domaines dans lesquels la technique l'emporte en efficacité sur la spontanéité.