Découvrir le travail avec un ordinateur

François Sass, Étienne Vandeput

DÉCOUVRIR LE TRAVAIL AVEC UN ORDINATEUR

F. SASS, E. VANDEPUT

Les élèves de l'enseignement secondaire ne sont pas allergiques à l'informatique. Que du contraire. Pourtant, l'algorithmique et la programmation ne semblent pas accessibles à tous. En Belgique francophone et depuis plusieurs mois, un vent souffle dans la direction de l'utilisation des logiciels. Professeurs, étudiants mais aussi parents pensent confusément qu'il y a dans ce domaine un plus grand intérêt pour l'informatique à exploiter.

Est-ce un vent favorable ? Que peut offrir l'école dans le cadre d'un tel enseignement et surtout faut-il enseigner l'utilisation des logiciels ? Si oui, ... quoi et comment ?

C'est à toutes ces questions que nous allons tenter de répondre. On pourrait d'ailleurs en ajouter une autre dont la réponse risque de déchirer les participants à un colloque sur la didactique de l'informatique :

L'utilisation des logiciels est-elle du domaine de la discipline (la science) informatique ?

Notre impression (un peu caricaturale, il est vrai) est que pour beaucoup d'informaticiens, tous les travaux dont la réalisation finale est prévue pour un ordinateur sont polarisés en travaux nobles (analyse et programmation) et travaux inconsistants (utilisation des logiciels).

Dans le cas de la programmation, il semble que l'exécution différée du travail ennoblisse celui-ci dans la mesure où l'ordinateur est accessoire, du moins dans un premier temps. Au contraire, utiliser un logiciel conçu pour un ordinateur d'un certain type pour un travail qui s'effectue le plus souvent en direct, est considéré comme une activité légère, dépourvue de toute démarche intellectuelle. De plus, certains logiciels touchent des domaines spécialisés nécessitant des connaissances voire des compétences externes à l'informatique ce qui peut laisser croire que l'on s'éloigne du domaine de l'informatique.

Cette dernière réflexion nous inspire un premier commentaire. L'informatique est une discipline qui ne peut vivre pour elle-même. L'informaticien ne peut se contenter de ses propres compétences, il doit être ouvert à d'autres disciplines pour pouvoir dialoguer. Par ailleurs, n'oublie-t-on pas un peu vite que les débuts de la programmation n'ont pas toujours été très nets en ce qui concerne les méthodes et qu'il a fallu du temps et de l'expérience pour que les idées s'éclaircissent à ce propos.

Les questions deviennent donc :

En quoi l'utilisation d'un logiciel peut-elle être méthodique ? Y a-t-il dans ce type d'activité des démarches analytiques ? Y a-t-il des concepts importants à cerner ? En quoi certains traitements différés rapprochent-ils l'utilisation des logiciels de l'analyse et de la programmation ?

Plutôt que de répondre systématiquement à toutes ces questions, voici quelques réflexions qui vous permettront sans doute d'y apporter vos réponses personnelles.

Trois points importants seront illustrés :

l'ordinateur comme **exécutant**

l'**information** et le **traitement formel**

l'**automatisation** des tâches.

L'informatique étant souvent définie comme le traitement automatique de l'information, les trois points qui précèdent peuvent constituer la base méthodologique d'un apprentissage raisonné. Ils sont présents à tout moment, lorsqu'il s'agit de donner une explication cohérente en cas d'échec d'une manipulation, mais aussi lorsqu'il s'agit de justifier une exigence syntaxique ou les limites des traitements possibles. Leur développement peut garantir dans des délais rapides une certaine autonomie de l'apprenti utilisateur.

L'ORDINATEUR COMME EXÉCUTANT

Cette pratique méthodologique s'inspire, il ne faut pas s'en cacher, de ce qui s'est déjà fait dans le domaine de la programmation. Le principe en est le suivant : l'ordinateur dont la mémoire centrale est chargée d'un système d'exploitation et d'un langage de programmation est un exécutant dont les compétences sont limitées à un certain nombre d'actions primitives. Dans ce cas, le but du jeu est d'écrire des marches à suivre dont l'exécution est différée. La connaissance des primitives de base permet un apprentissage élémentaire. L'apprenant peut alors, en fonction de ses besoins, découvrir d'autres primitives et ainsi dévoiler progressivement l'exécutant qu'il apprend à découvrir.

Pour un langage de type procédural, voici une liste de primitives qui permettent cet apprentissage élémentaire :

la lecture, l'écriture, l'affectation ;

les opérations mathématiques ;

quelques procédures et fonctions rapidement utiles ;

les structures de contrôle des actions...

Lors de la découverte de l'exécutant, l'étudiant est amené à se poser régulièrement la question : y a-t-il une primitive qui permet de faire faire ce traitement à l'exécutant ? Si la réponse est oui, le problème est résolu. Sinon, il est amené à écrire lui-même une procédure ou une fonction qui répond au problème. Celle-ci s'écrira au moyen des primitives existantes à moins que sa complexité ne nécessite une analyse descendante.

On peut tenir le même raisonnement si le langage de programmation est remplacé par un logiciel. Le plus souvent, le travail s'effectuera en direct (pilotage) sauf dans le cas d'écriture de macro-commandes ou si le logiciel permet la programmation. Lorsqu'un traitement doit être effectué par un ordinateur dont la mémoire centrale est chargée d'un logiciel, ou bien l'action est possible directement, ou bien il faut mentalement la décomposer en une série d'actions possibles. Très souvent, cette décomposition est séquentielle mais la nécessité d'analyser le traitement se fait sentir.

Voici quelques exemples choisis dans l'utilisation élémentaire de logiciels de types différents.

- Avec un traitement de texte :

  comment définir un bloc de texte ?

  comment permuter deux paragraphes ?

- Avec un tableur :

  comment définir un bloc de cellules ?

  comment déplacer un bloc de cellules ?

- Avec un gestionnaire de fichiers :

  comment sélectionner des fiches répondant un critère donné ?

  comment copier des fiches répondant à un critère donné dans un autre fichier de même structure ?

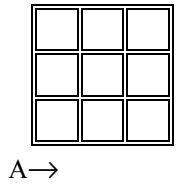
Remarquez que les réponses à ces questions peuvent être trouvées en tenant simplement compte de l'aspect formel du traitement et indépendamment du choix du logiciel. Toutefois, on constatera à l'utilisation que l'ordre des instructions dans la séquence est important et dépend des choix ergonomiques du concepteur.

Voici un exemple simple permettant de poser la problématique de l'exécutant et du faire faire :

Un Martien, qu'il faut guider, est capable d'effectuer deux actions. Les consignes pour les lui faire exécuter sont les suivantes :

Description de l'action :	Consigne pour son exécution :
avancer d'un mètre	prononcer le mot blanc
effectuer un quart de tour à gauche	prononcer le mot noir

La figure représente (vu d'en haut) un petit parterre de trois mètres sur trois. Les lignes marquent les allées dans lesquelles il doit obligatoirement se déplacer. Il se trouve au point A, orienté dans le sens de la flèche.



Il faut faire en sorte qu'il se rende au point B.

Que faut-il lui donner comme indications ?

La recherche de plusieurs réponses conduit à de nombreuses réflexions sur la manière d'analyser le problème, les contraintes syntaxiques, les exigences du faire faire, la notion de procédure... Elle montre aussi la nécessité d'avoir un plan en tête avant de se lancer à corps perdu dans l'énumération d'une série de commandes. Le plus important n'est-il pas d'abord de choisir un chemin, puis de vérifier si les actions primitives du Martien vont lui permettre de le suivre ?

Citons simplement l'exemple suivant : si on choisit de faire partir le Martien en commençant par une rotation à gauche, comment résoudra-t-on le problème au bout de l'allée, lorsqu'on voudra le faire tourner à droite ?

L'INFORMATION ET LE TRAITEMENT FORMEL

Les réflexions qui précèdent remuent un autre principe, semble-t-il incontournable en matière d'apprentissage. Le traitement de l'information par un ordinateur est purement formel. La définition de l'information, dans les traitements informatiques, doit être réduite à son aspect syntaxique, à l'aspect de sa forme. L'aspect sémantique est totalement absent bien que certains algorithmes bien étudiés puissent donner l'impression contraire. Tous les exécutants sont construits. Il en résulte que leur intelligence est artificielle, car limitée à l'exécution de traitements formels.

L'information est une suite de symboles dénuée de sens.

L'ordinateur est une machine à traiter les informations. Ces traitements sont limités aux traitements purement formels.

L'étudiant qui perçoit bien cette énorme limite du traitement informatique peut éviter de nombreux pièges et porter sur les produits qu'il utilise un regard critique quand à leur conception. Voici des exemples permettant d'illustrer la différence entre traitements formels et non formels.

Si on demande à une personne de remplacer le point d'interrogation par une lettre dans les deux cas suivants :

CH ?T CH ?EN

ça ne lui pose pas problème (pour autant qu'elle connaisse la langue française) et dans la mesure où elle donne à cette question une interprétation plus poussée que de simplement mettre une lettre, n'importe laquelle, à la place du point d'interrogation.

Pour trouver la réponse, cette personne (être humain) n'effectue pas un traitement formel.

Si on lui pose la même question, et si elle veut résoudre le problème de la même façon dans les cas suivants :

A ?GYAL O ?DOG

ça lui pose problème si elle ne connaît pas la langue dans laquelle ces mots ont du sens. Elle ne peut donner de réponse à moins de disposer d'un dictionnaire lexicographique de cette langue. Le traitement prendra alors plus de temps et la réponse résultera d'un traitement formel de ces informations.

Ces exemples permettent à l'étudiant de deviner les traitements possibles d'un correcteur orthographique et parfois, ses dysfonctionnements. Ils donnent une idée plus claire de la manière dont ces traitements sont effectués et battent en brèche l'idée d'ordinateur intelligent véhiculée par l'importance de certaines couches logicielles.

D'autres concepts auxquels il est habitué sont redéfinis dans le contexte informatique. En traitement de texte, on peut s'intéresser à la définition de mot, de phrase, de paragraphe, de page,... et faire découvrir le concept de délimiteur. Dans l'utilisation du tableur, il peut comprendre que la reconnaissance des types de données soit implicite (comme elle peut l'être dans certains langages de programmation). Avec le gestionnaire de fichiers il ne s'étonnera pas, dans une base de données, de l'absence d'enregistrement répondant au critère : nom = DUPONT alors qu'il est certain d'avoir encodé les renseignements concernant Monsieur Dupont...

Les exemples qui trouvent des explications dans le formalisme des traitements sont nombreux et quotidiens. Ainsi, pourquoi certains langages ou logiciels acceptent-ils des commandes écrites en majuscules ou en minuscules ?

On ne peut donc plus affirmer que l'algorithmique, du moins à un niveau de base, ne puisse aider l'utilisateur dans sa manière de concevoir son travail avec l'ordinateur.

Il sera à même de comprendre les exigences d'une syntaxe parfois rigoureuse et pourra apprécier le travail de fond du concepteur lorsque ces rigueurs syntaxiques ont été aplanies ?

Prétendre à un apprentissage efficace et raisonné de l'utilisation des logiciels sans jamais développer le concept de traitement formel, cela semble relever de l'exploit!

L'AUTOMATISATION DES TÂCHES

Ce troisième volet s'attache également à la méthodologie de l'apprentissage. L'ordinateur est choisi comme outil dans des domaines où l'automatisation apporte de nombreux avantages. Il serait stupide de l'oublier.

D'autre part, la finalité du travail est généralement une production sur support externe. Pour simplifier, considérons qu'il s'agit d'une imprimante. Cela signifie que l'utilisateur a toujours une idée plus ou moins précise du résultat escompté. Cette idée, proviendra souvent du souvenir, plus ou moins précis, qu'il a, d'une présentation particulièrement réussie d'un travail équivalent. En traitement de texte, il s'agira de la mise en page particulière d'un document ; avec le tableur, il s'agira d'un tableau au format précis ; avec le gestionnaire de fichiers, il s'agira d'une liste reprenant diverses données dans une présentation choisie ;...

A partir de l'examen de cette forme finale, les traitements essentiels peuvent être dégagés et correctement formulés. Une réflexion peut aussi avoir lieu sur l'avenir du travail produit : mise en page à modifier pour un texte (choix de styles,...), modification des données et simulations pour une feuille de calcul, choix d'une structure adéquate liée à un questionnement optimal de la base de donnée dans le cas d'un gestionnaire de fichiers...

Ce souci d'une automatisation à outrance du travail réalisé nécessite une démarche particulière qu'il est bon de développer dès le début. Ainsi, sans vouloir parler de paradigmes, chaque logiciel, quelle que soit sa catégorie va influer sur le mode de raisonnement de l'utilisateur qui cherche à en tirer un parti maximum.

Voici un petit exemple qui illustre ces derniers propos dans le contexte de la découverte d'un tableur :

Après une enquête effectuée auprès des élèves d'une classe, on a voulu dresser un tableau des résultats. On souhaite une présentation semblable à celle qui suit.

Comment peut-on mener la réflexion sur le travail à faire faire ? Quels sont les concepts de base qu'induit un exemple aussi simple que celui-là ?

MOYENS DE TRANSPORT	NOMBRE D'ELEVES	FREQUENCES
VELO	7	0,23
PIED	6	0,20
BUS	10	0,33
VOITURE	2	0,07
MOBYLETTE	5	0,17
	30	

Devant la nécessité d'une telle réalisation, se pose d'abord la question du choix de l'outil. Un traitement de texte conviendrait bien dans la mesure où le document ne doit pas être reproduit avec d'autres données. Cependant, pour dresser ce tableau, certains calculs doivent être effectués. Si on dispose d'une calculette, le travail peut se faire sans effort.

Si on est amené à dresser plusieurs documents de ce type, avec des données différentes, le traitement de texte n'est plus l'outil idéal. On peut imaginer l'emploi d'un gestionnaire de base de données. Chaque enregistrement contiendrait les

données nécessaires au calcul des autres informations (ici, les fréquences et l'effectif total).

L'outil le mieux adapté semble cependant être le tableur car la production finale s'apparente à une feuille de calcul. On y trouve des chaînes de caractères, des valeurs numériques données et des valeurs numériques calculées. Si le choix se porte sur ce type de logiciel, on peut, toujours en observant le résultat final souhaité, déterminer les informations correspondant à chacune de ces catégories d'informations. D'autres questions se posent. Par exemple, les moyens de transport seront-ils toujours les mêmes ? Le tableau sera-t-il complété dans la suite par l'apparition d'autres informations calculées ? Dressera-t-on un ou plusieurs graphiques illustrant la situation ?

L'exemple choisi n'est pas très complexe mais il montre que le problème de l'automatisation se pose très tôt à l'utilisateur. Dans l'optique d'un apprentissage raisonné et efficace, on ne peut le négliger, voire l'aborder en fin de cycle de formation. C'est un état d'esprit qu'il faut susciter au plus vite.

Que penser de l'utilisateur qui crée une base de données sans avoir au préalable pensé sa structure en fonction des interrogations qu'il sera amené à porter ? La rédaction d'un texte demande que ce texte soit pensé, qu'il soit écrit, qu'il soit bien présenté. L'utilisateur peut-il raisonnablement réfléchir au contenu de son texte en l'encodant et en se préoccupant de sa mise en page. Ne vaut-il pas mieux qu'il apprenne à séparer ces différentes phases pour en améliorer la qualité (création de feuilles de style et de macro-commandes avant toute mise en page sauvage, par exemple) ? N'y a-t-il pas, finalement, beaucoup de choses à apprendre aux étudiants dans un domaine où beaucoup de gens semblent ignorer qu'il peut exister des démarches efficaces ?

F. SASS

FNESeC, rue Guimard, 1
B-1040 Bruxelles

Tél : 32 2 5070755
Fax : 32 2 5070746

E. VANDEPUT

Inspection diocésaine
Namur Luxembourg
Chaussée de Charleroi, 193
B - 5070 Vitrival

REFERENCES

Mode d'emploi des logiciels

WordPerfect 5.1, Word, Ventura
Dbase III+, Clipper 5.0
Excell, Quattro, Lotus
MSDOS, Windows, Apple System

Livres sur la mise en page

Manuel de typographie et de mise en page : Fr. Richaudeau
Mise en page : R. Parker, L. Thérien
Maquette et mise en page : Duplan et Jauneau
DEsign Principles for desktop Publishers : T. Lichty

Livres sur la programmation et associé

Proverbes de programmation : Ledgard
Programmation structurée : Dijkstra
Programmer : Duchâteau
Algorithmes et structures de données : Wirth
Pascal : Wirth
Algorithm, Sorting, Seminumerical algorithm : Knuth
Machines à penser : Arsac
Pour une informatique consciente : Fontoliet
User centred system design : Norman, Draper
Human-Computer Interface Design Guidelines : C. Martin "Lin" Brown