

Programmation.

L'ordre quintilien impose à tout langage de respecter des principes de sobriété de vocabulaire, de pertinence des mots. Cela condamne tous ces langages où rien ne peut être écrit sans une lourde documentation, qui vous rappelle en permanence tous les détails d'un verbe et des mots qui s'y associent, sans lesquels le compilateur n'aboutira jamais.

Avec un tel environnement, le programmeur ne peut réfléchir à l'algorithme qu'il doit traduire, mais doit chercher en permanence les mots qui traduiront le plus efficacement ce qu'il veut "dire", à la machine qui doit réaliser.

Il est comme ces apprentis d'une langue étrangère : au début ils sont incapables de bien s'exprimer, car ils ne peuvent penser ce qu'ils veulent dire avec les mots de la langue en cours d'apprentissage.

Quelques explications.

Un langage bien conçu quand au vocabulaire, parle à l'esprit de l'artisan programmeur. Illustrons avec Windows.

Windows offre pour ses formulaires de dialogues un grand nombre de contrôles; ils n'apparaissent que dans ces dialogues; ils ont un nom, un code de repérage, un code du dialogue où ils apparaissent. Ils se représentent par un dessin d'objet, des paramètres de base qui l'habillent, le dimensionnent, le positionnent géographiquement dans l'espace du dialogue, le situent comme récepteur d'information ou donneur d'information ou les deux. Et des comportements additifs concernent les informations reçues ou transmises : ordonnées ou non, sélection simple ou multiple,...

Citons les objets : LABEL, BUTTON, LISTBOX, TEXTBOX etc. Ainsi décrits, la programmation devient claire, car les critères retenus pour l'objet sont faciles à mémoriser pour n'importe quel artisan. De plus il relira facilement le texte écrit, dès lors que l'on sépare la commande DIALOGUE, de celle qui exploite les Objets activés dans le dialogue, de celles qui enchaînent le travail de divers DIALOGUES. On peut ici penser au modèle PERT.

On retrouve la structure d'un programme Windows, mais sans l'obligation de passer par le cycle des messages que Windows analyse et redistribue, sous prétexte de bien contrôler le multitâche. De fait, Windows procède à l'espionnage permanent de ce que font les programmes : il ne cesse de pouvoir analyser les messages qui circulent. Cela a été vu avec l'analyse du projet Echelon.

Soulignons un aspect important : si nous créons un vocabulaire pour les objets liés à un espace, ce vocabulaire doit être unique. Si l'on réutilise ce vocabulaire pour des objets proches des précédents mais membres d'un autre espace, alors il faut leurs ajouter des qualificatifs, des attributs linguistiques pour souligner les différences. De telle sorte que l'on croit gérer une langue riche et précise, alors que l'on gère une langue de plus en plus imprécise et cacophonique.

Pour applications.

Pour atteindre ces objectifs, nous avons repris quelques principes qui furent à la base solide de l'informatique du software, dès sa naissance :

1- Usage d'un pré-compilateur éventuel : il permet de passer du langage de l'utilisateur au langage reconnu par le compilateur utilisé. Cela pour éviter l'écriture d'un compilateur dédié.

2- Usage d'un Compilateur connu et courant : FORTRAN. COBOL sont les plus adaptés, mais révisés dans le sens où PHP a été conçu. Compilateurs soutenus par l'Assembleur si nécessaire.

POWERBASIC est un compilateur puissant et permet le test de gros projets. On constate que : DLL, RC, LIB, COM, ne sont pas nécessaires. Il suffit de programmer différemment. Cela facilite bien des choses, moyennant des binaires chargés de 30 ou 50 kilos bytes en plus. Insignifiant pour les PC actuels. Mais on peut réduire tout cela.

3- Usage de l'éditeur de liens pour un compilateur classique : qui accepte les bibliothèques au format COFF ou OMF.

4- Usage de bibliothèques classiques (DLL, RC, COM sont éliminés). Le scientifique qui aspire à la maîtrise du modèle programmé, approchera toujours plus le modèle Quintilien d'espace temps. Son programme est proche d'une pure codification, expression du fonctionnement d'un objet réel. Une fois mis au point, ce n'est qu'un objet qui accepte des informations (toujours de même nature) et restitue un résultat (toujours de même nature). Il n'y aura rien à ajouter, rien à retirer. S'il peut appartenir à une classe d'objet, ce n'est qu'au même titre que ce mode de classement existe en sciences naturelles. En ce domaine, les éditeurs ont voulu copier une science du classement, sans connaître le génie des objets naturels dans lesquels cette discipline vise de mettre de l'ordre.

5- Usage d'un intégrateur au système d'exploitation, de bibliothèques permanentes. Lorsqu'un objet est d'un usage courant, par exemple : DIALOGUE, LABEL, BUTTON, LISTBOX, TEXTBOX etc, il doit devenir membre d'une bibliothèque système. On peut alors l'utiliser comme on utilise une commande DOS, même facilité, même puissance, même stabilité. La programmation en est allégée, facilité.

Ces brefs aperçus guident tout le travail de développement d'une Informatique. Celle qui devrait soutenir le processus de l'évolution de système d'objets minéraux ou organiques. Cette évolution, est et sera toujours un processus aux normes d'un Quintilien fait de tâches LST, et de celle du concept d'Unicité LMT.

Cloud Computing.

Par rapport à ce "nouveau concept", ce qui précède semble d'un autre temps. Cela vient de ce que l'Informatique dont on parle d'abondance, ne concerne que

la Bureautique, le son, l'image, l'animation, en bref la dimension publicitaire et commerciale de nos sociétés.

Tôt ou tard, mais sans doute trop tard, l'Informatique qui touche les activités non commerciales de l'humain artisan deviendra nécessaire. Cette informatique là sera toujours classique.

On pourrait qualifier le "cloud computing" comme le web électronique des centres de calculs d'IBM et autres des années 70.

Mais aucun artisan ne confierait les secrets de ses tâches à un gestionnaire de méthode de conduite de projet, de même, aucun ne confiera ses softwares éprouvés à des centres qui les mettraient en exploitation moyennant rémunération.

Il pourrait le faire pour des versions réduites par rapport à celles qu'il utilise pour son travail. En clair le Cloud Computing doit savoir respecter et surtout protéger la propriété privée du Savoir faire acquis.

Enfin on néglige tous les retards d'exploitation. qui viennent déjà des pannes des télétransmissions. Sur ce point, le cloud computing s'éloigne de beaucoup du principe d'Unicité LMT.

Il y a beaucoup, beaucoup à faire !