

Internet, le Web.

Rappel et exploitation de ce qui a été dit.

En 1971, chez Philips Data System, je fus chargé de traduire des programmes Basic en Fortran. Je devais opérer à Paris, les fichiers se trouvaient à Lille. Aussi a-t-on utilisé la ligne téléphonique pour transmettre ces fichiers. Je les recevais, je les traduisais, les testais avec les données transmises, dans les deux versions pour vérifier que les résultats étaient identiques. Cela fait, je les retournais. Deux ou trois aller retour suffisaient.

Se dégage divers aspects :

- La ligne téléphonique, matérielle ici, sera remplacée par la ligne de télétransmission. À chaque bout il y a un télétype lié à un ordinateur, aujourd'hui ce télétype est un écran.

- Aux deux bouts il y a un ordinateur sur lequel tourne, dans ce cas, le même programme, mais pas forcément le même système d'exploitation. Peu importe qu'il soit écrit en deux langages différents, mais proches. Il n'est pas nécessaire que les ordinateurs soient identiques, il leur suffit d'avoir les compilateurs des langages utilisés.

- Ces programmes faisaient déjà de la programmation dynamique : on saisit des données, on les traite, on restitue un ou des résultats, en les mettant en une forme lisible sur le papier du télétype. En 1971, certains centres des télécommunications seront déjà dotés d'écrans faisant du traitement de texte ligne à ligne. La ligne représentait la carte perforée de 80 colonnes.

Puis une remarque de fond :

Nous nous mettons dans la même situation. Mais nous allons opérer différemment.

Deux programmes P et P' tournent sur deux ordinateurs distants.

P est chargé de récolter des informations locales ou venant d'un autre ordinateur, de leur faire subir un traitement local si l'on veut, de les enregistrer, puis de les expédier à P' qui possède les algorithmes des traitements complexes.

P' les traite, élabore des résultats, les enregistre et les réexpédie à P.

P les reçoit et les met en forme pour l'utilisateur local qui les visualise sur son écran.

Cela avait un avantage considérable :

- L'utilisation de la bibliothèque de programmes faits, testés et d'usage public, enregistrés dans les diverses bibliothèques d'IBM et autres grands de ce temps.

- L'utilisation et l'affinement des trois langages de base FORTRAN, COBOL, BASIC et de leur support commun : l'Assembleur.

- Les systèmes d'exploitation, notamment celui d'IBM avec son concept de

Job et de Tâche, étaient prêts pour cela. Multitraitement et multitâche étaient déjà opérationnels sur les ordinateurs qui pouvaient Servir de nombreux utilisateurs.

Si l'on avait persévéré dans le sens du scénario décrit entre Paris et Lille, et cela était possible, nous ne serions pas dans une tour de Babel informatique. Les informaticiens qui ont perdu le sens des réalités, passent leur temps à copier et traduire les choses imaginées par un autre.

Les grandes inventions de l'informatique sont de ce passé où Fortran jouait le rôle, au niveau du cerveau, du rôle que jouait la Légion comme cadre naturel et matériel. Tout cela échappe totalement à l'Informatique 2000, qui a peu à voir avec celle de ceux qui ont réellement fondé cette discipline.

J'ai été curieux d'expérimenter quelques langages, en plus des 4 de base ci-dessus cités : C, Pascal, Simula, Algol, Lisp, Javascript, Vbscript, PHP, HTML.

Je suis arrivé à la conclusion que les inventeurs de ces langages n'avaient pas le réflexe cartésien. Je n'ai jamais eu de problème de passage de paramètres d'un programme appelant vers un sous programme appelé, nul besoin du prototype de la routine appelée. Mais, au besoin un fichier vocabulaire, indiquant le mode d'appel de la routine est préférable et plus efficace. On peut, pour le développement, y ajouter un fichier méthode, donnant les précisions pertinentes sur l'algorithme (la méthode) qui réalise le travail de la routine appelée.

Pour in fine, décider de n'utiliser que ceux des 4 langages de base dont j'ai besoin et que je connais. Mais je les utilise, ou tente de les utiliser comme cela se faisait encore en 1974. Je renonce aux librairies DLL, y trouvant plus d'inconvénients que d'avantages. J'élimine les fichiers de ressources RC. Tout doit pouvoir se trouver dans une librairie classique LIB, qui répond aux normes Run Time. J'utilise HTML d'une manière statique, le moins possible dynamique, puisque ce rôle ressort du scénario précédemment décrit pour Paris-Lille. Donc je ne me mêle pas d'écrire des CGI.

Lorsque les serveurs actuels exécuteront les applications .EXE d'un développeur de Site, comme le faisaient les systèmes d'exploitation des années 70, toutes ces nouveautés seront inutiles. Le bon site web, sera toujours fait par le bon analyste programmeur, qui aura aussi le sens du visuel, de la représentation pertinente de ce qui est essentiel. Mais tout cela nécessite un NET à faible taux de panne, à vitesse de transmission équivalente à celle d'un système matériel logiciel local, et revoir la note sur le "cloud computing".

L'Informatique suit une voie qui conduit les éditeurs à faire tous pratiquement la même chose s'il veulent traiter le même sujet. Tout cela arrivera à une impasse : l'ordinateur passera plus de temps au travail de conversion des données reçues, qu'à celui du traitement de ces données, pour atteindre le résultat pour lequel elles ont été soumises. Cela équivaut à produire une paralysie lente du système d'information, un blocage de l'évolution du système au sens de

l'évolution quintilienne.

Si l'on veut débloquent le système, cela imposera que l'on s'en retourne au scénario Paris-Lille illustré ci-dessus. Ce n'est qu'une question de temps.