Pour un ISBN de l'informatique

Yves Trémolet

L'organisation: {lecteurs, maisons d'éditions, auteurs, références bibliographiques, bibliothèques, librairies, acheteurs} aurait probablement à l'heure actuelle du mal à fonctionner sans l'espace des références ISBN (International Standard Book Number). Cet article, que l'on pourrait résumer par quelqu'un peut-il me dire pourquoi l'informatique, mère des catalogues, voue-t-elle une haine aux références ? (les siennes) voudrait dire que l'informatique et les télécoms ne couperons pas à l'existence à priori d'un ou quelques espaces de ce *type* , qu'une certaine dynamique réside dans l'horizontale, ou que réalisation du concept de système informatique ouvert et existence de tels espaces sont des synonymes.

Modèle

Nous présentons ci-dessous un modèle permettant de décrire les «inscriptibles» concepts passés, présents, et futurs, guillemets ou pas, choisissez:

Ce modèle, sorte de lambda calcul sans notion d'exécution ou le header généralisé est basé sur la syntaxe suivante:

*connaissance -> concept**
concept -> concept-id contexte-id définition-du-concept

C'est à dire, la connaissance est vue comme une liste de concepts (ou ensemble, l'ordre n'ayant ici pas d'importance mais il faut l'écrire). Chaque concept a un nom (ou un numéro), ainsi qu'une définition exprimée dans un certain contexte, ce contexte étant lui même un autre concept. La définition sera vue comme une chaîne de symboles dans le premier alphabet à partir duquel on puisse extraire un sens, un alphabet à deux éléments, bien qu'un message dans un alphabet à un élément véhicule toujours le nombre de signes mais alors il n'y a plus du tout de grammaire (séparation).

Nous avons comme d'habitude un modèle cyclique. La récursivité, héritage des tautologies logiques, est omniprésente en informatique: à l'intérieur des langages, les métalangages se décrivant eux-mêmes, les compilateurs souvent écrits avec le langage qu'ils compilent, les recherches en sémantique formelle cherchant des langages dont la sémantique puisse se décrire en utilisant ces mêmes langages, le métamodèle d'une base de donnée étant décrit avec le langage de cette base, la définition d'un microprocesseur sur un ordinateur tournant avec ne posant qu'un problème temporel (dans des écritures). En allant plus loin, dans la mesure ou un symbole ne peut se décrire que par l'intermédiaire d'autres symboles, on voit mal comment éviter ces cycles et «La science est un cercle qui se referme sur lui-même, un cercle de cercle» (Hegel, science de la logique), encore que cercles, lignes, points, question d'approche; science, réalité expérimentale, écriture de, question, ou ici hors sujet (ce qui n'est pas en contradiction avec la citation ci-dessus). Cette propension «naturelle» qu'ont les modèles (ou leurs écritures) à se refermer sur eux mêmes, pourrait d'ailleurs expliquer le mal qu'ont les systèmes informatiques à devenir ouverts, ceci sans même entrer dans des considérations plus stratégiques.

Un exemple de boucle de démarrage

Le tableau ci-dessous représente un exemple de démarrage de la base proposée:
La première colonne contient le numéro du concept, la deuxième le numéro du contexte, et la troisième la définition du concept. Ce tableau qui est à priori une chaîne de bits au format F2 est passé par des transformations qui devraient apparaître claires en le lisant.

Les concepts utilisés comme contextes de définition sont 93 et 95:

- Le concept 93, appelé CODE_DEF, est un langage de description de codes ou alphabets sur nombre de bits constants.

  L'alphabet ASCII a été choisi pour sa syntaxe.

  Ce concept est utilisé comme contexte pour définir l'alphabet ASCII.

- Le concept 95 est appelé texte ASCII. Une définition utilisant ce concept comme contexte de définition doit être interprétée comme un texte informel dans l'alphabet ASCII. Ce concept est en particulier utilisé pour se définir lui-même.

0	95	la base elle même une représentation concrète (écriture) de cette base existe comme une chaîne de bits au format F2 dont le présent texte en est une copie.
1	95	zéro: le symbole 0, note: nous n'avons pas ici écrit une définition graphique des symboles mais rien n'empêche
2	95	un: le symbole 1
3	95	bit: le bit: l'alphabet {0,1} un bit: un symbole de l'alphabet bit
4	95	l'octet: le langage constitué des chaînes de 8 symboles sur l'alphabet bit un octet: une chaîne du langage l'octet. les octets: l'alphabet constitué par l'ensemble des chaînes du langage octet
5	95	l'octet null: 00000000
6	95	le format F2: (<number1> null <number2> null <number3> null <bitstring>)* où, null est l'octet null. <number1>, <number2> et <number3> sont des chaînes d'octets à sémantique ASCII représentant chacune un entier naturel en notation décimale (ces chaînes ne peuvent pas contenir l'octet null par définition). <number1> est le numéro du concept, <number2> le numéro du contexte, et <number3> est le nombre de bits que contient <bitstring>. <bitstring> est la définition du concept <number1> dans le contexte <number2>. Le format F2 fourni donc un langage concret de représentation de la base.
7	95	le symbole 2
8	95	le symbole 3
...	...	....
14	95	le symbole 9
15	95	le symbole a minuscule
16	95	le symbole b minuscule
17	95	le symbole c minuscule
...	...	...
36	95	le symbole z minuscule
37	95	le symbole A majuscule
39	95	le symbole B majuscule
62	95	le symbole Z majuscule
63	95	CR: le retour chariot, retour à la ligne
		Autres symboles ou concepts (comme CR) de l'alphabet ASCII

93	95	CODE_DEF : langage de description d'alphabets ou codes sur nombre fixe de bits Une définition utilisant CODE_DEF comme contexte doit être interprétée comme une chaîne de caractères ASCII ayant la syntaxe ci-dessous: --------------- 93:CODE_DEF <nom_du_code> <nb_de_bits> <bitstring_val> <concept_number> ..... <bitstring_val> <concept_number> --------------- CODE_DEF est un "self identifying format", SIF (96) La sémantique d'une définition utilisant le format CODE_DEF est la suivante: <nom_du_code> est un identificateur représentant le nom du code. <nb_de_bits> est le nombre de bits en notation décimale constituant un caractère du code décrit. Les couples <bitstring_val> <concept_number> décrivent les associations (valeur de la chaîne de bits)->interprétation bitstring_val est un entier en notation décimale. <concept_number> est un numéro de concept en notation décimale (correspondant en général à un symbole).
94	93	93:CODE_DEF ASCII 8 ... 48 0 49 1 50 2 ... 57 9 ... 97 10 98 11 ... 122 38 ...
95	95	texte ASCII: une chaîne de bits interprétée dans le contexte 95 est une chaîne d'octets interprétée avec la sémantique du code ASCII comme un texte informel.
96	95	SIF: self identifying format: un format dont la premiere ligne est de la forme: <concept-number>:<nom du format>, ceci en ascii, les associations <concept-number>-<nom du format> sont décrites par le concept 97 (name_map) remarquons que la définition d'un tel format exige de connaitre son nom et numéro à priori, peut-être la raison pour laquelle de telles choses existent si peu.
97	95	name_map: ceci est un espace de noms ou une fonction associant un numéro de concept à une chaîne ASCII, une valeur de cette table peut-être trouvée par le concept 98 (non représenté ici)

L'effort de formalisation n'a pas été poussé très loin dans cet exemple. En effet, l'idée n'est pas ici de tenter le bootstrap formel étranger à toute vrai nouveauté ou réécriture et, les mots ou signes (dialogue et sang) restent toujours dans/autour les atomes et lignes de fronts, cependant, en tant que chaîne de bits, cette description a le côté formel conféré à toute écriture symbolique de part la possibilité de faire des copies (c'est ça et pas autre chose), même si deux copies d'une même disquette n'ont rien de plus secrètement parfait que deux copies d'un même livre, ou même machine à vapeur. L'intérêt d'une telle base ou espace de références résiderait premièrement dans son existence, c'est à dire dans la fourniture des numéros, l'accent est ici mis sur le côté séparateur et non sur le côté descriptif.

Une définition pourrait être quelque chose comme :

ceci est le langage de programmation L2. Pour sa définition , vous pouvez consulter le livre:
"L2, the language" ISBN : 0123456789

En fait, on pourrait purement et simplement retirer le côté définition, considérer les numéros en tant que connecteurs potentiels et ramener le modèle à: c'est à dire rien. La définition peut donc être vue comme une trace de dépôt ou retrait de numéro, une première publication au sujet du concept contenant des informations tout autant administratives et juridiques que techniques. Remarquons également que l'on peut voir le concept 0 de deux manières différentes: Comme une fonction retournant une définition pour un concept donné ou comme le texte constitué par une écriture de la base. En effet, les images de la fonction devant être écrites, la fonction ne peut être définie qu'en extension. Cependant, rien n'empêche de déposer un autre concept correspondant également à la base mais retournant des définitions différentes sur les mêmes références. Enfin, il n'est pas interdit d'appeler base une instance de copie ne contenant pas toutes les définitions, la base n'a pas besoin d'exister de manière centralisée (cas de tout système d'exploitation et publications associées).

OS ou système d'exploitation

A ce sujet, citons "Inside Macintosh":

> Every application must have a unique signature by which the finder can identify it. The signature can be any four-character sequence not being used for another application on any currently mounted volume (except that it can't be one of the standard resource types). To ensure uniqueness on all volumes, you must register your application's signature by writing to:
>
> Macintosh Technical Support
> Mail Stop 3-T
> Apple Computer, Inc.
> .....
>
> Note: There's no need to register your own resource types, since they'll usually exist only in your own applications or documents.
>
> .....
> Like signatures, file types must be registered with Macintosh Technical Support to ensure uniqueness. When the user chooses Open from an application's File menu, the application will display (via the Standard File Package) the names of all files of a given type or types, regardless of which application created the files. ....
>
> Note: Signatures and file types may be strange, unreadable combinations of characters;
> they're never seen by end users of Macintosh.

Si la facilité d'utilisation du Macintosh est avant tout due à son interface graphique ainsi qu'à sa grammaire (aux concepteurs de ces concepts), elle est également indissociable de cette possibilité d'identification unique. Ceci permet des liens automatiques types de fichiers/types d'éditeurs, des procédures d'installation /désinstallation génériques, et surtout offre au système la possibilité de suivre les rangements de publications référencées effectués par l'utilisateur sur son disque dur ou le réseau. Ces espaces plats sont en général partagés en trois parties: une pour Apple, une pour le reste du monde, et une pour tricher au niveau local, local ayant ici le sens: ‹‹je sais (suppose) que toute chaîne lue par le présent programme a été écrite par ce même programme ou un autre que je connais, j'ai donc toute latitude quant à son interprétation dans le présent programme››». Il est actuellement difficile de déménager d'une *installation* UNIX vers une autre et les incohérences de "paths" ou de nommage sont souvent plus à la source des problèmes que les types de machines ou versions du système, la hiérarchie UNIX hésitant entre *modèle de rangement* et *espace de référence*, (ou refusant de considérer l'espace de référence qui s'est construit grâce au modèle de rangement, il suffit d'utiliser un modèle de rangement partout de la même manière pour créer un espace de référence), amalgame entre besoins de rangement et de nommage en un point toujours inévitable mais également au nœud des problèmes d'installation ou d'évolution. Certains nouveaux concepts d'UNIX tentent d'ailleurs d'y

remédier par l'intermédiaire d'autres espaces de noms par exemple IMAKE le MAKE du MIT à l'aide de variables d'environnements (dont les noms n'ont eux rien de variable).

D'une certaine manière, un système d'exploitation peut se voir comme l'ensemble des adresses (ou références) des concepts utilisant ce système et des concepts du système, une boucle cette fois *formelle* (en soi, puisque ça tourne) de définitions sur silice plastique ou magnétique[1] offrant un certains nombres de portes pour augmenter la boucle. Ces références peuvent apparaître de manière plus ou moins précises: elles nécessitent une interaction dans le monde Macintosh (et dans beaucoup d'autres) et apparaissent sous forme de conventions (le premier qui publie une écriture utilisant la référence) dans des mondes comme UNIX (bien qu'il y ait aussi sous UNIX l'espace des `magic numbers` du fichier `/etc/magic` identifiant des types de fichiers (ou d'architectures machines), espace géré par interactions (en principe)). Un sondage est actuellement effectué sur le net (Internet, Information Highway) au sujet du suffixe utilisé pour un programme C++ ceci dans un cadre incluant aussi bien UNIX , VMS, MSDOS ou WINDOWS (ce suffixe est important pour les concepts qui l'utilisent comme identificateurs, comme par exemple les formalismes de makefile). Une autre caractéristique de ces références est qu'elles ne sont pas recompilables. Le monde constitué par un système d'exploitation pourrait en ce sens être caractérisé comme un interpréteur que l'on n'arrête jamais et où, à un certain niveau, le ramassage des miettes (garbage collection, réutilisation de références) est impossible pour maintenir la compatibilité ascendante, ou doit se faire à la main si on est sur que plus personne utilise un concept (son identifiant), tout ceci n'étant pas très net puisque l'on peut aussi tricher (par exemple: this publication is subject to change without prior notification).

Mais tout ceci existe déjà

Tout d'abord, puisque nous parlons de numéros, n'oublions pas que la puissance de l'exponentielle (nombre de numéros sur taille d'un numéro) est pour une fois dans notre camp. N'est il pas étonnant que tout les livres publiés tiennent dans dix malheureux chiffres (en fait neuf le dixième correspond à un procédé de vérification des neufs premiers). Ceci va même plus loin que les livres, puisque si vous achetez une carte de Canton à la sortie de la gare elle a également un numéro. Il y a donc vraiment beaucoup de numéros.

Les espaces croissants de numéros existent dans des domaines divers et variés: La numérotation ISBN est peut-être la plus impressionnante, (du fait que l'organisation nécessaire à leur distribution ait été mise en place plus que par leur potentialité d'existence), bien qu'il y ait aussi les espaces associés aux autres codes barres ou EAN-GENCODs. On pourrait séparer ces espaces entre «espaces ouverts» et «espaces fermés» dans le sens ou soit l'organisation générant les numéros est également responsable de leurs sémantiques (publications associées), soit elle a surtout un rôle de

---

[1] A propos de harware/software, remarquons que la définition de quasiment toute pièce «harware»(disons de concepts dont la définition, l'écriture, est directment graphique) peut aujourd'hui se retrouver sur une disquette et qu'inversement, point intéressant, la définition de logiciels sous forme graphique est souvent perçue comme le nouvel Eldorado, ramenant les problèmes de layout des fils propre au hardware qui lui essaie plutôt d'aller dans l'autre sens (je crois). Note: Je veux ici parler des formalismes graphiques mais pas des outils permettant de manipuler des pièces logicielles à fonction graphiques «en tant que telles», comme par exemple les outils de design d'interfaces. Si il y a une différence, c'est peut-être qu'à l'execution, avoir une pièce logicielle ou sa référence revient quasiment au même bien qu'avec l'automatique ... et, au niveau étiquettes, aucune différence:

```
>Does anyone know how to get a unique keyboard identifier from an X
>server in general?
There is no neat and general way, but a kludgy way which we use here is to do a
'xmodmap -pk' and then compare the result with stored maps and the
match gives the keyboard type. However this requires that the server gives
a unique mapping for every keyboard and the unpatched server from MIT
doesn't do this for type5 sun keyboards. However with my patch applied it works.
```

Si ce n'est que pour pouvoir «coller» une étiquette sur un logiciel il faut avoir auparavant prévu la place. Le logiciel se révélant ici plus dur que le matériel (si l'on tient à introduire une typologie entre ces diverses publications).

distribution de connecteurs ubiquitaires dans un cadre pré défini, unification conceptuelle/synchronisation symbolique (tierce personne). Dans le contexte de l'informatique et des télécoms, citons les espaces de l'ISO (celui-ci n'est en rien limité à l'informatique et aux télécoms), du CCITT (maintenant ITU-T), les RFCs («Request For Comments» documents définissant les normes du monde Internet, par ailleurs modèle d'horizontalité. La première date de 68 et la numéro 1 de 69), et beaucoup d'autres. Dans le contexte du modèle OSI, il y a aussi le concept d'Object-ID. Un Object-ID est une liste de numéros pouvant référencer n'importe quel concept, body X400, attribut X500, type de modem ou encore organisation ... Ces listes sont organisées selon la hiérarchie des organisations qui les définissent, le top level appartient à l'OSI et au CCITT sous l'égide de l'ONU et de nombreuses organisations font partie de cette hiérarchie. Si vous avez un début de liste, vous pouvez l'étendre à votre guise. Nous sommes donc ici dans la problématique structure organisationnelle, structure typologique, ou pas de structure du tout d'un espace d'identifiants.

Problèmes d'analogie.

Les livres ont l'avantage qu'une fois qu'ils sont écrits et que leur publications leur a associé un numéro, ce numéro identifie un objet unique (un ensemble d'objets identiques), deux éditions d'un même livre ayant deux numéros différents tout comme les éditions des traductions d'un même livre.

Pour ce qui est d'un concept informatique (ou télématique), il est souvent plus difficile de créer une telle association signifiant/signifié ou identifiant/écriture. Un programme devant être écrit dans un certain langage et environnement, il ne peut être utilisé que si il est connecté à d'autres concepts et nécessite donc souvent de nombreuses éditions qui peuvent également avoir des numéros de versions. D'autre part, le but étant que les autres concepts l'utilisant ne fasse pas la différence entre les différentes éditions, donner un nom différent à chaque édition ne résout pas le problème. D'une certaine manière, on a autant besoin de référencer le programme, librairie ou langage, en tant que concept, idée, titre ou mot qu'en tant que publication.

En plus, un nouveau concept peut lors de sa définition, utiliser plusieurs des portes précédemment ouvertes, sa définition devenant contextuelle. Ou, une librairie A dans un langage X peut souvent être utilisée dans un langage Y, la librairie A n'est donc pas dans X mais à côté, elle aurait pu être écrite dans le langage Y, et une définition peut également être vue comme l'utilisation dans les autres définitions de son identifiant. Dans le cas des formats ou langages de programmation il est encore plus problématique d'isoler une définition dans le système mais il n'empêche que des identifiants (étiquettes qui peuvent aussi valser) sont nécessaires pour pouvoir créer des indirections (évitons ici les «paradoxes» de dialogues, fuite du message externe, ou empilement notationnel, il s'agit de publications écrites.)

Ne sommes nous pas ici dans le sens du mot ouverture ? Ce mot a souvent dans le contexte OSI le sens suivant: Permettre à plusieurs auteurs ou constructeurs de fournir différentes écritures pour un même concept (souvent défini par ces même acteurs), ceci se traduisant par exemple par connecter ce concept à des concepts différents dans divers contextes. Mais, ce mot est souvent utilisé en informatique avec un sens différent: fournir un concept-contexte permettant à de nouveaux concepts ou nouveaux concept-contextes de venir s'y accrocher (sens existant d'ailleurs également dans le contexte OSI). Ce deuxième sens est typique de l'évolution d'UNIX puisque le sens du mot UNIX a plus ou moins évolué en incluant les nouveaux concepts venus s'y accrocher (mail, lex, yacc, filtres de compressions/décompressions, shells, sockets, perl, X window(plutôt moins que plus dans ce cas mais les autres concepts sont aussi apparus dans d'autre systèmes)....). UNIX montre donc par là, si il en était besoin, qu'un modèle (système) peut se développer dans un environnement de publication. Le premier sens correspond à ouverture vue de l'utilisateur (possibilité d'acheter la «même» chose chez divers fournisseurs) le deuxième vue du concepteur, possibilité de modifier une chose existante directement sans au préalable devoir encapsuler la chose dans un nouveau contexte (rôle de tierce personne de l'OS). Ajoutons ici un troisième sens correspondant à la facilité d'accès aux définitions, exemple:

```
> Help !! Where can I get an official technical specification (or less
> formal ones) of the RTF (Rich Text Format).

You can get version 1.2 of the specification from ftp.microsoft.com,
but it's available in more formats on ftp.primate.wisc.edu, which
also has software for dealing with RTF files.  The current software
distribution is 1.09, which is afflicted with the inability to understand
the myriad new symbols invented for version 1.2 of the spec.  Release
1.10 of the software should be ready in a few days and is not subject
to this problem.
```
(discussion sur le newsgroup `comp.text`)

Et, à l'heure actuelle, il est souvent difficile de dissocier les interfaces ou spécifications des programmes qui les sous-tendent. Par exemple, dans le cas des API de fenêtrage ou des interfaces systèmes, l'écriture de l'interface (le manuel de référence de la librairie) exige en général l'écriture du programme pour être capable de faire «tourner» la définition et donc de la déboguer (ou de la définir). Ceci n'exclut pas l'existence de plusieurs publications pour un même concept mais il reste le fait qu'une norme devient en télématique le manuel ou notice d'utilisation d'un produit rendant le produit parfaitement transparent si il respecte la norme (sans doute vrai dans d'autres domaines). Vue de l'utilisateur, une norme n'est plus une contrainte sur le produit mais devient le produit lui-même. (quand ça n'est pas le texte de la norme qui peut être directement utilisé par les machines, exemple ASCII ci-dessus). Inversement, tout produit ou notation est potentiellement une norme.

Cette analogie soulève un autre problème plus lié à «Le fond-et-la forme» ou forme est ici entendu dans un sens très restrictif (à l'opposé de style) comme la partie mathématique statique d'une écriture de fond: l'alphabet (sans les signes, en gardant le nombre de signes) pour les textes qui en utilisent (et encore, espaces, fontes, italiques, indices exposants, tout ceci susceptible de retomber dans la base statique.). A propos des symboles, autre problème d'analogie puisque il s'agit ici de recenser ceux qui existent. Notons à ce sujet l'immense travail effectué par le consortium UNICODE et/ou l'ISO à travers ISO10646, recensant tous les symboles utilisés couramment dans des espaces de 16 ou 32 bits. Où l'on apprend aussi, dans le domaine des codes de caractère, en lisant la mailing list consacrée à ces normes, que la ligature œ (celle de cœur, sœur, nœud, bœuf, œuf, œil, vœu, œsophage, chœur, œdipe) serait capable de créer des incidents technico-diplomatiques, étant exclue du code ISO-latin-1 devant être le standard courant sur 8 bits pour les écritures utilisant l'alphabet Latin[2] et variantes, pas très grave la preuve, et question signes j'entérine.

Tous les concepts informatiques cités jusqu'à présent sont des concepts de formes, des définitions de média ou de contenants ou modes d'interactions (de lecture) avec ces concepts. De ce fait, le problème est tout autant d'arriver à des consensus que d'en définir de nouveaux. Remarquons cependant que certains programmes informatiques peuvent être considérés comme étant de l'information de fond. Certains programmes informatiques ont d'ailleurs déjà un numéro ISBN, c'est par exemple le cas de la version "pile hypercard" du dictionnaire IEEE??? décrivant les termes couramment utilisés en informatique, les livres enregistrés sur cassettes ont également des numéros, les jeux électroniques sont aussi de l'information de fond. Plus les représentations deviennent symboliques, par exemple passer d'un format de FAX à Postscript, plus les critères de définitions du format sont liés au futurs contenus, la forme gagne sur le fond ou la forme des uns devient le fond des autres. Ces formats de «haut niveau» nécessitent un partage plus grand de symboles ou identificateurs entre émetteurs et récepteurs. Le format de fax ne nécessite que le partage du codage des pixels, mais un format comme Postscript nécessite le partage de noms de fontes, de noms de types de formes graphiques, de noms de couleurs , ou modèle de définitions de couleurs... En ‹‹remontant›› encore, par exemple dans les formats de type SGML, on partage la notion de chapitre, de table des matières, de préface, etc. SGML est un bon exemple de la dualité (utilisation générique)/(modèle statique vide), puisque sa définition, devant la complexité de la tâche , a amené à définir plus un «méta»langage qu'un langage, en reportant alors le problème de normalisation sur des programmes

[2] Ajoutons que les codes correspondant pour les alphabet Arabes ou Cyrilliques s'appellent aussi ISO-latin-??, (ou iso-8859-??), la raison étant que les noms sont "parallèles" au object-id et que l'on a pas accroché ces alphabets au top level mais simplement l'ensemble de ces alphabets(je crois).

écrit en SGML, programmes décrivant un certain type de document devenant à leur tour des normes (la notion de préface n'est donc pas écrite dans SGML mais dans beaucoup de programmes écrits en SGML, la définition de tout langage ou format amène toujours à des questions du type: mettons nous ceci dans le langage ou dans une librairie ?). Ceci a l'avantage d'ouvrir le concept, mais amène également au fait que la réalisation d'éditeurs (interfaces homme machine) pour le concept statique acceptant n'importe quel paramétrage est quasiment impossible ou devient dépendante d'un paramétrage particulier.

Les formats de données, concepts de représentations des connaissances[3], sciences humaines génératives, solfèges, existent dans des domaines divers et variés et n'ont pas attendues les machines: Textes (nombreuses approches), langage de description de pages, formats d'images, 2D, 3D (nombreuses approches), de circuits VLSI, de son, d'écriture musicale, de jeux électroniques, de machines (langage de programmation, nombreuses approches), de commandes, de factures, de relevés pétroliers, de formules chimiques, de lettres (les signes), de dossier médical, de carrosseries, de mondes virtuels, d'architecture, d'images animées , de textures, etc.

Note:
(
Sans s'étendre sur l'utilisation actuelle du terme virtuel, il y a — c'est certain —, glissement ou switch sémantique propice à jeux de mots vaseux par rapport à:
«L'œuvre pure implique la disparition élocutoire du poète, qui cède l'initiative aux mots, par le heurt de leur inégalités mobilisés; ils s'allument de reflets réciproques comme une virtuelle traînée de feux sur des pierreries,..» (Mallarmé)
ou:
«dites, le dictionnaire me suffirait : soit, trempez-le de vie, que je devrai en exprimer pour employer les termes en leur sens virtuel» (idem)
ou encore:
«Je sens sous ma pensée le terrain qui s'effrite, et j'en suis amené à envisager les termes que j'emploie sans l'appui de leur sens intime, de leur substratum personnel. Et même mieux que cela, le point par où ce substratum semble se relier à ma vie me devient tout à coup étrangement sensible, et virtuel.» (Antonin Artaud)
On entend même parler de communautés virtuelles où encore d'interviews virtuels sous prétextes qu'ils sont conduits par E-Mail... Il est vrai que ce mot a depuis longtemps glissé en Informatique: mémoire virtuelle, ou mêmes fontes ou fonctions virtuelles, ceci correspondant en général à une indirection supplémentaire dans un espace d'identifiants avec éventuellement une opération entre les deux. Pas grave, vocabulaire technique, on fait ce que l'on veut, il y a aussi le théorème des travaux virtuels en mécanique (plus compréhensible dans ce cas), mais peut-être pas une raison pour VIOLER le sens commun[4] (même si ça ferait rire aux éclats n'importe quelle infante): musée virtuel ou même musée imaginaire ! à propos d'un serveur World Wide Web, ceci venant du ministère de la culture, pitié je deviens fou, toute action de copie parfaite devrait être strictement interdite[5], mesure de salut

[3]On entend souvent qu'il y a des problèmes dans la représentation des connaissances , cependant, lorsque l'on sait ce que l'on veut représenter, en général on y arrive. Traduction: l'intelligence artificielle, comprise comme volonté d'écrire la capacité à scruter le zénith, dont le discours m'a toujours attendri ou pompé l'air, me paraît vouée à l'échec d'après l'énoncé du problème, (je ne comprend pas ce que ça veut dire) pardon à l'ambulance, mais l'histoire; désir actuellement égaré dans certains bavardage cognitifs (citations en réserve). Et puis remplacer "en" par "de" dans la dénomination académique "recherche en intelligence artificielle" ne serait pas plus mal, mais alors plus trés académique.

[4] Sens commun qui ne se limite d'ailleurs en rien à celui des dictionnaires. Ou alors expliquez moi pourquoi virtuel est toujours présenté sous forme d'une imbécile opposition à réel («En somme, se dit Durtal, malgré les dissidences de quelques-uns de ses textes, la cathédrale est lisible»). Mais si l'on considère qu'un dictionnaire, tente de miroiter une chose mouvante en phase de négociation permanente n'ayant de réalité qu'en chaque individu ou cervelle, parler d'un sens commun est alors virtuel au sens qui n'est pas dans les dictionnaires. Ecriture Prométhéenne, on voudrait maintenant en nier l'existence (ou le rapport lecteur auteur (même si anonyme)).

[5] Il ne s'agit pas ici de savoir en quoi une copie est parfaite, mais des capacités de diffusion.

public, le vocabulaire informatique est vicié jusqu'à la moelle, je défie quiconque de m'expliquer: niveau conceptuel, niveau logique, niveau physique, si ce n'est en me présentant trois syntaxes et en me disant: celle-ci est conceptuelle, celle ci est logique, celle-ci est physique (Ah bon).
Bien que l'emploi «grand public» de virtuel venant de l'informatique a démarré avec "virtual reality" aujourd'hui réduite à VR où là, l'association posant une question annonce la couleur. Mais assister au raz de marée dans la presse ou l'édition: société virtuelle, démocratie virtuelle... est tout simplement effrayant, commençons par interdire toute utilisation frauduleuse du terme virtuel.

On pourrait continuer avec le terme «Images de synthèse», en disant par exemple que les techniques (et modèles mathématiques) utilisés pour définir une lettre (le signe) ou une carrosserie peuvent être strictement identiques (le nombre de dimensions n'étant souvent qu'un paramètre en ce qui concerne les modèles mathématiques), et que ceci se fait souvent à la main (avec une souris):
«
- La modélisation tridimensionnelle est la seule compatible avec les exigences d'un processus de conception de forme complexes en ce sens qu'elle permet de répondre interactivement à des requêtes du type:
"Si je déforme cette ligne, quelles vont être les répercussions sur l'ensemble du volume ?"
» (M. Pommellet, «La modélisation tridimensionnelle en conception de navires»)
)

Si la numérisation permet à priori de limiter le nombre de médiums et protocoles de base nécessaires, elle ouvre donc la porte à une multitude de formes de —publications — (« «Multimédia» is the wrong word says MIT's Negroponte. «Everything has now become digitized,» He says «We have created a unimedia, really. Bits are bits» » (Newsweek Mai 93)).

A toutes ces formes de publications, il faudrait également ajouter la multitude de modes d'interactions qui leurs sont associés: outils de création, de consultation, d'échange, de stockage (et les identifiants de catalogues associés), une publication pouvant également contenir son mode d'interaction comme par exemple un jeu électronique (à un ou plusieurs, au même endroit ou pas).

Ces formes se développent de plus en plus par composition, on met du postscript dans du TeX, du TeX, du CGM ou du POSTSCRIPT dans du SGML ou HTML, et du SGML du POSTSCRIPT du TeX ou du CGM dans un système de fichiers. Mettre le système de fichier dans du SGML est à priori également possible tout comme simuler le processeur 486DX au dessus de TCP/IP ou X400 éventuellement à partir de la disquette destinée au outils de microgravures. Le développement par composition étant également vrai pour les outils gérant ces formes, filtres et éditeurs «communicant» de plus en plus («»: abus de langage signifiant ici que les textes définissant les filtres et éditeurs contiennent de nombreuses références conceptographiques réciproques (mots clés) ou signes de consensus entre leurs concepteurs).

L'expérience réside dans ces domaines notationnels dans l'utilisation et la vérité dans la survie. Pour que l'expérience ait lieu, il faut donc qu'un concept puisse se «diffuser» dans ceux qui existent, c'est à dire par exemple qu'un format ait une référence dans les espaces plus ou moins isomorphes de types de fichiers, de messages dans les protocoles d'E-mail ou de drag and drop, ou d'étiquette typologique dans les langages à typage dynamique tel smalltalk, tout comme un livre (son identifiant), une fois qu'il a un numéro, est virtuellement (sens potentiel) présent dans tous les systèmes de gestion des librairies et bibliothèques. Le fait est qu'à l'heure actuelle, un concepteur de formes doit aller s'inscrire dans tout ces espaces ou un gérant d'espace doit gérer ce qui existe, problème classique du guichet unique mais qui ne sera sans doute pas résolu en définissant ce que doivent être les messages mais plus en permettant au concepteur de concept-contexte de reboucler les portes ouvertes sur des sources existantes —lors de leurs définitions — et en assumant les top level de fait créer au niveau des systèmes de fichiers ou protocoles de messageries, (il existe aussi des formats de chorégraphie (Merce Cunnigham, Antenne 2 Juillet 93, aussi Le Monde 25/26 Juin 95), il y a donc vraiment beaucoup de formats). Rebouclage pouvant se faire directement ou, si cela est jugé déraisonnable (sans doute moins souvent qu'on ne le pense), en déposant un nouvel espace de noms

dans le premier, mais si le concepteur a quelques doutes sur l'ouverture des sources existantes, il est probable qu'il préfère la gérer lui-même.

La pression de consensus existe à tout les niveaux cependant, dans le cas du logiciel, la dynamique de constitution du consensus est peut être différente du fait que le coût de production est nul, bien que le degré d'interdépendance créé dans le logiciel et surtout le fait que beaucoup de concepts logiciels émergent au niveau de l'utilisateur (et sont donc appris d'une manière ou d'une autre par l'utilisateur) jouent certainement dans l'autre sens. Par coût de production nul, nous voulons dire ici le coût nécessaire à faire des copies, (ce qui permet par exemple à la Free Software foundation, muni d'un réseau de distribution tel qu'Internet ou CD en librairies d'être leader dans certains secteur de marché (défini comme ensemble de consommateurs)). Ce sens coût nécessaire à faire des copies, est apparu au XIX^{e} (Alain Rey (dictionnaire historique de la langue française)) lors de la révolution industrielle. Du fait que ce coût est nul pour le logiciel, le sens a glissé en revenant plutôt à produire du logiciel ayant le sens produire des plans. Remarquons également que dans les autres domaines ou le coût de production (sens copies) est faible par rapport au coût de développement, par exemple disques ou cinéma, production a là un sens financier, sens qui existe déjà pour le logiciel dans le contexte des jeux électroniques. En ce qui concerne la télévision (ou pixels sur écrans cathodiques) production a aussi le sens financier bien que là, pour des raisons techniques, le mode d'interaction a démarré dans la diffusion synchrone en direct ou différé (peut-être le plus virtuel que l'on puisse imaginer), mode d'interaction actuellement en révolution tout d'abord dans les structures potentielles émetteurs/récepteurs. Dans le domaine juridique, ou le but est plutôt de trouver des choses qui se sont produites, production a le sens de montrer: produire les pièces à conviction. Tout cela pour ? Dire que ce glissement est responsable d'amalgame permanent en informatique entre faire quelque chose et faire des copies, en particulier dans les cercles de philologie systémique (cycle de vie du logiciel), philologie qui mène au crime d'après Ionesco, ou des théories ayant trait à un cas sont transportées sans encombre dans le second, ou par exemple dans l'acronyme ESF pour European Software Factory (à moins qu'il s'agisse d'une référence à la New Yorkaise). N'oublions pas aussi que beaucoup de logiciels existent en un seul exemplaire, la production de copies se ramenant alors à un état des lieux. Ce coût nul a d'autres conséquences, par exemple le fait que la frontière recherche/industrie devient très floue, ceci étant d'autant plus vrai que les résultats fondamentaux en informatique ont souvent un caractère d'impossibilité (problème de l'arrêt, non existence du lambda calcul typé sans restriction ...), et l'écriture d'un programme peut être vue comme l'écriture d'un théorème.

Des noms ou des nombres ?

Essayons de ne pas entrer dans la problématique de l'existence ou non des nombres, problématique qui n'a ici pas forcément lieu d'être puisque nous parlons pas de nombres mais de numéros, il est vrai homonyme en Anglais bien qu'il y ait # ainsi que l'inversion syntaxique phone number, number of phones, le nombre est lui omniprésent dans la structure des définitions.

Le fait est qu'il est difficile d'aboutir à un consensus large sur la syntaxe et la taille maximale d'un identificateur (pourquoi d'ailleurs essayer ?). Ceci peut être illustré par cet extrait tiré de /TeX3.14/README:

```
Remember that write-white devices take special care.
          The Ricoh 4080 print engine is one such device;
          try the RicohFourZeroEightZero mode_def;
          and see the file README.WRITE-W for more information.
```

Anecdotique à l'unité mais multipliable au fini dans les écritures actuelles.

Le problème de la représentation existe aussi bien sur aussi avec les numéros, (il ne s'agit jamais que de chaînes de caractères), exemple (réunion Internet: comp.protocol.tcp-ip)

```
In article <2366@mefos.se> pam@mefos.se (PA Monwall) writes:
>Is there a way to map the ethernet card address to a hostname or
>internet adress without running a setup program at each host ?
>We have a host running SCO Unix which reports the error:
>"rwhod: malformed host name from c047b764"
                                 ^^^^^^^^^
>My guess is that the number is a ethernet adress but I don't know
>what host it is.

      That looks to me like an IP address in hex, without the dot separators. An
ethernet address is 6 bytes, (2 hexs digits each).
That is 4 bytes, just like an IP address. Converting hex to decimal and putting it
into dot format will make it clearer.
```

On utilise des numéros tout en «bas» par exemple pour les symboles, quand on ne peut pas faire autrement, et tout en «haut», pour nommer les normes, la raison étant là qu'un consensus à priori sur un espace utilisant des noms est problématique. Entre les deux, cela varie, mais il existe de nombreux contextes ou l'on utilise des noms (mots clés) par exemple dans les «mondes» constitués par les langages de programmation. La plupart des langages définissent des «entités de publications» (librairies, packages, clusters, modules).

Si la conception d'un langage de programmation peut être considérée comme un problème scientifique ou mathématique, son utilisation introduit toujours des problèmes qui sont eux organisationnels (rappelons ici que le langage Pascal dans sa première version ne permettait pas d'écrire un programme en plusieurs morceaux). La convention utilisée est en général celle d'un «top level» *de fait*, celui qui correspond aux actions de publications[6]. C'est à dire, on écrit "#include <X11/X11.h>" mais pas "#include </US/MIT/X11/X11.h>" (optique organisationnelle) ni "#include </prog_languages/c/lib/X11/X11.h>" (optique typologique) ou "#include </home/logiciels/gui/packages/X11/X11.h>"(optique rangement local). Cependant, ces espaces doivent eux exister d'une manière ou d'une autre dans le système (/lib ou /usr/lib ou /usr/local/lib dans le cas d'UNIX).
Liens en dur ! Liens en dur ! cablé en dur ! Quoi ? Il en faut, principe même de tout plug and play, le code barre d'un compact disc est aussi écrit sur le disque lui-même (ainsi parfois que sa table des matières), exemple:

[6]Ceci ne veut pas dire que tout doit être au top level, le langage pouvant à travers sa syntaxe introduire la notion de référence locale (ce qui existe en C), c'est à dire d'autres top level dont les procédés d'indirections n'ont eux rien de local.

```
I am also interested in acquiring the following info from the audio CD:

* Time for each track. hh:mm:ss resolution is sufficient.
* If availiable: Album title, song title and song artist.
* If availiable: EAN code, Record Co., publication date
* Any other info stored on the Audio CD.

I am also interested in reading the whole RAW image off the CD and placing it
on a (yes, quite large) hard disk for manipulation.
[...]
I have (an old) specification of the MSCDEX api, but i do not have a
specification of the calls MSCDEX uses to call the hardware drivers. Does
someone have such a spec?

Is there anyplace i can get the read and orange book specifications for free,
maybe as text files?
[...]
NOTICE: This information will not be used to copy CDs (my request might suggest
that) but rather to make the procedure of filing new records at a radio station
2000% more efficient.
```

Il est plus urgent de donner aux concepteurs de langages ou formats la possibilité de déposer des «espaces de noms» (ou de correspondance noms<->numéros vers des espaces existants) sans savoir ce que cela veut dire (en évitant les fuites) plutôt que d'essayer de définir des «méta» modèles du type PCTE (portable common tools environment) ou le problème des connecteurs arrive toujours après, identifiants d'«espaces de noms» pouvant par la suite être utilisé par des modèles du type PCTE ou tout autre modèle de rangement existants.

Le numéro peut être vu comme pseudo signifié ou signifiant tautologique, carburant idéogrammatique jetable, distribuable par paquet grâce au modèle numérique, permettant de garantir aux signifiants courants la «souplesse» d'emploi ou context sensitivness nécessaire, pouvoir changer le concept qu'il y a derrière un nom ou le nom qu'il y a devant un concept, dessus dessous, dessus dessous, capacité de dissociation atomique souvent utile, fond-et-la-forme. De plus, nombre des identifiants dont on parle ici finissent toujours pas émerger au niveau de l'utilisateur final, utiliser des noms comme pivots amène alors au fait que les tables de correspondances n'étant pas nécessaires par défaut, elles ne voient jamais le jour. Et ce n'est pas parce qu'il y a des numéros que l'utilisateur les voit même dans le cas particulier ou il programme. Par exemple, le concept de socket sous UNIX , bon exemple de nœud conceptuel, définit par l'intermédiaire du fichier "socket.h", une correspondance noms<->numéros pour 19 protocoles. Plusieurs langages ont un mot clé `obslolete,` permettant, lorsqu'il est associé à une structure, d'avertir les utilisateurs de la structure que celle ci est considérée comme obsolète. Cependant, le nom de la dite structure se retrouve alors ‹‹consommé›› (dans le contexte de structure de publication du langage). Pour pouvoir ne pas consommer d'identifiants textuels, il faut pouvoir ramener la consommation sur un autre espace. Dans le contexte des langages de programmation, on peut imaginer des schémas ou l'évolution d'une librairie ne consommerait que des numéros, sans que personne ne les voit (en général), et sans casser la compatibilité ascendante même dans le cas de librairies partagées. On parle sur le net de «semantic real estate» ou encore de «name space pollution».

Avoir à la fois des noms et des numéros permet aussi qu'il y ait plusieurs numéros pour un même nom correspondant à des définitions contextuelles mais gérées au même niveau dans un contexte englobant. Les librairies étant de plus en plus partagées, leurs références deviennent plus «dures» et ces problèmes se posent de manière effective dans la pratique. C'est ce qui se passe automatiquement à travers le polymorphisme introduit par les langages objets ou fonctionnels.

Il est cependant certain que le fait d'utiliser uniquement des identifiants textuels confère une certaine dynamique, en permettant par exemple à des concepts tels que le «README» de se mettre en place presque comme dans le langage, là ou une approche normalisatrice aurait probablement définie une kyrielle d'attributs, et à ces attributs de se «cristalliser» ensuite à l'intérieur du README. Cependant, ces identifiants deviennent vite des mots clés (utilisés par d'autres programmes), perdent alors toute possibilité d'évolution et paragraphe suivant. A propos du README, notons, et c'est le cas pour de nombreux autres fichiers, que le symbole README tout comme "man" ou "bin" fait partie

intégrante du modèle UNIX tel qu'il est utilisé, ces symboles sont au "vrai" root d'UNIX, ce système comme beaucoup d'autres, permettant à un fichier de passer du statut «marchandise étiquetée» à «valeur d'un concept du système dans un certain contexte», ce qui est vraie de tout les systèmes qui survivent, il suffit de l'écrire puis de le considérer, (pas de vraie frontière modèle/données).

Espace plat ou autre

Remarque préliminaire:
Quand le contexte d'utilisation d'un espace d'identifiants conceptuels (correspondant à des publications) grandit, la structure de cet espace à plutôt tendance à s'aplatir que le contraire. Ceci est typique dans le cas des documents. L'espace des références ISBN est plat (la structure interne des numéros peut être vue comme un schéma de distribution et semble d'ailleurs variable (à part le côté pays)) alors que dans beaucoup d'organisations de moindre taille, les références documentaires sont beaucoup plus longues typées et compliquées. Dans le domaine des codes barres, il est probable qu'un aspirateur puisse se retrouver à côté d'une boîte d'allumette, table de repassage ou de dissection. Ceci tendrait à montrer que quand l'on peut s'arranger on a plus tendance à mélanger rangement et nommage, ou qu'une indépendance des acteurs conduit à un consensus moins grand mais plus précis favorisant en retour l'interconnexion des objets produits par ces acteurs c'est à dire l'écriture des bibliographies. De la même manière, l'espace des part numbers IBM est plus plat que n'importe quel "home directory", et il est souvent plus facile de trouver quelque chose dans l'espace des serveurs FTP (qui eux mêmes s'aplatissent) de l'Internet que dans l'environnement local ou l'on se trouve, même si cette chose a déjà été ramenée. Enfin, selon une approche mathématique, si ces concepts (écritures) sont différents, rien n'empêche de les numéroter et donc de les ramener au même niveau, toute réflexion à ce sujet ne pouvant en éviter la potentialité.

Espace plat ne veut pas dire absence de structure, rien n'empêche de déposer les concepts de structure, ensemble, ou chemin d'accès ou correspondances noms<->numéros dans le même espace, le fichier «rfc-index» (listant toutes les RFC publiée jusqu'à présent) ou encore la RFC "Internet assigned numbers" publiées régulièrement sous diverses références pourraient aussi avoir un numéro de RFC, il y a d'ailleurs de tel document dans le contexte du CCITT. Si l'informatique offre un avantage, c'est bien de pouvoir — plus facilement — considérer de manière statique des concepts statiques mais à contenu dynamique (services), le 11 du Minitel, 3615 METEO ou encore les FAQ (frequently asked questions lists de l'internet).

Il y a cependant une dualité temporelle: contrainte administrative de naissance (donc temporelle) mais diffusion automatique / moins de contraintes de naissance mais diffusion problématique.

Pourquoi les espaces de références hiérarchiques ne marchent pas ?
Il se pose toujours, au moment de la publication, le problème de savoir «ou va t'on l'accrocher», ceci se terminant souvent par, on l'accroche là, là, là et là. Et, à l'époque des réseaux de consortiums, start up et développeurs indépendants ou universitaires (il est certainement plus facile pour un consortium de publier un livre qu'un «object» au sens OSI, le rôle de l'éditeur est là, il est vrai, important (il se reflète dans la référence ISBN)), ce problème se pose en permanence (déjà au niveau 2 puisqu'il y a une branche «joint-iso-ccitt»). Et surtout, avec un espace hiérarchique, il est impossible de jeter remplacer, «oublier», un concept de structure typologique ou organisationnel sans jeter avec les objets contenus ou devoir les renuméroter ou réidentifier. — Si les éléments sont identifiés dans le même espace que l'ensemble —, l'identifiant de l'ensemble ne constitue alors qu'un chemin d'accès pouvant être remplacé par un autre sans renuméroter les éléments. D'une certaine manière, au bout d'un moment, les espaces hiérarchiques ne se remplissent plus (et se nettoient encore moins).

Exemples:

```
Claudio Nieder <claudio.nieder@alcatel.ch> wrote:
```

>Hi,
>
>I've come along an object identifier which starts with
>
>   {joint-iso-ccitt(2) 16}
>
>X.208 only gives the main arcs below the ccitt(0) and iso(1) trees. Does anyone
>have an idea where the main arcs below joint-iso-ccitt(2) are described?

Steedman (in ASN.1 Tutorial & Reference) lists a few (but not 16)

0	presentation layer
1	ASN.1
2	acse
3	reliable-transfer
4	remote-operations
5	directory
6	mhs
7	document-architecture
8	transaction-processing
9	management

ou:

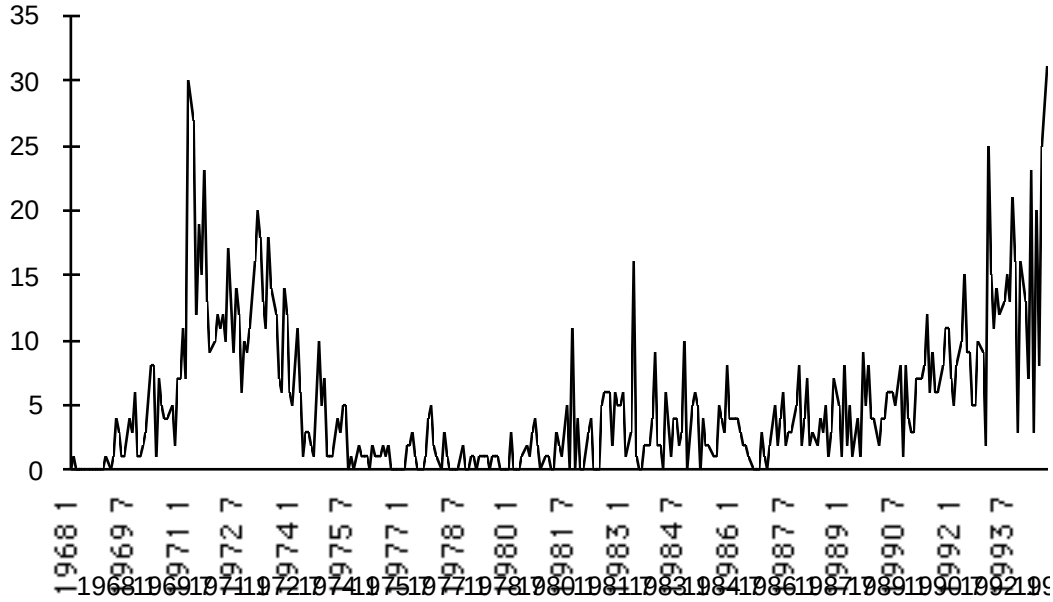
Examples: Multi-national OR-names, DNIC-s, object-identifier
arcs. Ericsson Telecom have asked ITU-T to assign them multi-
national object-identifier arcs, and ITU-T has refused,
saying that Ericsson must ask for this from Sweden.

The discussion applied to assignment new identifiers either
only for governmental international organizations (like the
United Nations etc.) multi-national organizations of
companies (like SITA, European Railway Association, etc.),
and multi-national companies (like Ericsson Telecom, IBM,
etc.).

People said that it was dissatisfying that CCITT during the
whole four-year previous study period did not solve this
problem in any ways.

(note de réunion de jpalme@dsv.su.se postée sur comp.protocols.iso)

Rythme quadri annuel à comparer au nombre de RFC publiées par mois:



En voulant créer des «espaces sémantiques» sans synonymes conceptuels (concepts différents mais ayant plus ou moins la même utilité) on se retrouve avec cette fois de vrais synonymes (plusieurs identifiants pour le même concept). Nous parlons ici de références à longue durée de vie, ces «liens symboliques» apparaissent souvent à travers la documentation, il suffit de l'écrire et de le considérer.

Un modèle hiérarchique est certainement nécessaire à «l'intérieur» des publications, pour l'expression des définitions, par exemple, le concept d'entité de publication d'ADA est passé d'un modèle plat à un modèle hiérarchique lors de la transition ADA->ADA9X, mais ne marche pas au niveau des publications, la granularité des publications constitue le choix de packaging permanent par rapport à la dualité temporelle précitée, bien que l'on puisse, après, repackager plusieurs publications dans une seule (exemple: /ISBN/2-13-045142X = {/ISBN/2-13-045139X, /ISBN/2-13-0451403, /ISBN/2-13-0451411}).

Et puis le problème n'est pas là, le problème est: l'Horreur de la ligne :
«Straight line is the downfall of mankind» (in the «Beauty Of Fractals»),
ou plutôt:
«et il n'est pas de pointe plus acérée que celle de l'Infini» , (Baudelaire, Le Spleen de Paris, 3).
ou:
«I could be bounded in a nut-shell, and count myself a King of infinite space; were it not that I have bad dreams» (Shakespeare, Hamlet, 2 2)

Ici associée à l'effroyable banalité de l'ubiquité parfaite réalisée ainsi qu'à l'angoisse de la consommation irrévocable (difficile).

Mais dans ces histoires d'écritures, le meilleur moyen de ne jamais voir la ligne (ou de passer son temps à la reconstruire), est qu'elle existe.

Et : Il n'y a pas d'identifiant, il n'y a que des connecteurs. Un numéro ISBN peut identifier un livre, son prix, son auteur, la liste des bibliothèques qui l'ont, rayon où il se trouve dans une certaine bibliothèque, liste de ses lecteurs, etc ... Il suffit de l'écrire et de le considérer. De la même manière, un mot clé peut identifier un format, un outil de création pour ce format, une librairie de chargement pour ce format, etc ... (ce qui n'empêche pas la librairie d'avoir par ailleurs un autre identifiant.)

Proposons ISCN pour "International Standard Connecting Number". Il y a là des choix entre identifier quelque chose avec un nouvel identifiant ou par composition d'identifiants pré-existants, espèce de bataille face à la combinatoire, ou plutôt utilisation de la combinatoire pour limiter le nombre d'identifiants créés.

On dit souvent que les systèmes formels sont fermés, dans le sens ou ils définissent une forme dans laquelle doit venir se loger toute écriture utilisant le système. Cependant, il apparait en pratique qu'il n'en est rien car: on relie en permanence des écritures dans un système A à des écritures dans un système B, à travers des mots clés, on écrit en permanence des filtres ou traducteurs d'un système formel dans un autre, les mots clés créant les connexions n'étant ni dans un système ni dans l'autre, un modèle "englobant" (et non unique) est toujours en retard, même si il suffit de l'écrire, et, — vue de l'écriture — , on pourraît dire que le sens de ces mots clés évoluent en fonction des connexions qu'ils permettent de créer.

Donc, il n'y a ici rien à démontrer, les systèmes s'écrivent avec les espaces de noms à leurs dispositions, mais ils s'écrivent de plus en plus de manière itérative (ou par sédimentations, il paraît qu'il peut y avoir 30 millions de lignes de code dans un commutateur téléphonique, effets pervers du cout de production nul), pour les écrire, il faut créer des connexions et l'expérience montre que les espaces plats sont ceux qui passe le mieux à l'échelle ou même, que plus l'échelle est grande, mieux ils passent à l'échelle. La réalisation de ces espaces, obligeant à se placer «au-dessus» des différents acteurs (ou plutôt: au centre), implique elle un consensus sur un schéma de distribution entre ces acteurs, ceci se traduisant souvent par une structure à l'intérieur des numéros. Il ne s'agit même pas de savoir si il faut des espaces hiérarchiques ou autre mais que les top level doivent être ouverts, on peut toujours accrocher n'importe quel modèle hiérachique sur n'importe quel top level. Il s'agit de constituer des *axes de références* en eux-mêmes porteurs de *dynamique*, cette histoire de mélanger classification et nommage est une maladie (le dictionnaire de l'académie n'a résisté qu'à une publication non alphabétique, (préface de ce dictionnaire)).

Toute nouvelle syntaxe ou file-system peut réarchitecturer ‹‹le root›› (et le reste) en introduisant le nombre de niveaux d'indirections souhaités au cas où, un peu comme un réseau téléphonique privé introduit les indirections voulues vers l'extérieur. Le fait que tout réseau privé utilise le même préfixe pour sortir n'a pas grande importance par rapport au fait que les numéros soient partagés «à l'extérieur». En continuant dans un parallèle espace d'instances/espace de concepts, chaque téléphone doit pouvoir appeler chaque téléphone, la structure des adresses a donc un effet direct sur le système, elle participe au fonctionnement du système. Dans le cas des concepts (ou ensembles de copies), si la définition de tout concept peut à priori utiliser n'importe quel autre, on est plus ici dans le cas "cartes éthernets", c'est à dire qu'une numérotation plate dans un contexte global permet la constitution de n'importe quelle conceptographie (réseau local dans le cas ethernet).

Il se produit cette chose étrange: dans le cas des instances, par exemple numéros de téléphone, une numérotation plate paraît «naturelle», puisque l'on parle de choses d'un certain type bien établi. Mais cette numérotation participant souvent au fonctionnement du système, à travers les tables de routages, on se retrouve parfois avec des espaces trop plats, exemple actuel, l'espace des identifiants sous le domaine ".com" des adresses E-mail internet. Par contre, dans le cas des concepts, d'une certaine manière, les tables de routages sont constituées par l'écriture même du système, les connexions sont toujours "en dur", la structure de l'espace ne pose donc pas les mêmes contraintes de fonctionnement. Et là, par contre, les structures sont en général sur-compliquées. Remarquons qu'il n'y a pas de vraie frontière entre instances et concepts, une adresse ethernet peut également désigner un certain type d'imprimante (ou l'ensemble des imprimantes de ce type sur un certain réseau), ou même un "pur" concept (se réduisant à sa référence, symbole), comme les adresses de broadcasting (ou désignant tout ce qui est branché sur un certain réseau). Déjà vrai avec le téléphone à travers 12 ou 911, ce problème se retrouve partout, par exemple dans les espaces d'identifiants des processus de systèmes d'exploitation.

Sans doute pas un drame de ne pas mettre la lettre A dans le même espace que X11 , X400, l'organisation ISO, un modèle de modem ou de machine à laver, norme de représentation de nombre en virgule flottante, la Chine, la France ou l'Angleterre, (une certaine tempérance mystique est, dans

ces histoires d'écritures, de rigueur) cependant, prétendre à un avantage quelconque hormis des questions de taille de représentation (souvent importante[7], associé au caractère statique ou dynamique) est une fiction, le contraire est écrit en permanence, exemple:

```
Stable implementation
agreements for open systems
interconnection protocols
part 8 - message handling systems

june 94

ftp: nemo.ncsl.nist.gov/pub/oiw/agreements/08S_9406.ps

Section content types:

there are presently no object identifiers for content types allocated by the X400
SIG.  (Cette norme a 10 ans !!)

It is recommended that the application provider (organization) use a hierarchical
structure for identifying their data types to ease the administration of the
identifiers.

For example, company PCSoftware Inc, obtains the organization number "999" from
ANSI. The PCSoftware SpreadSheet file for MS-DOS might be assigned the following
object identifier:

{Iso (1) member-body (2) US (840) PCSoftware Inc (999) MS-DOS-Application (1)
 SpreadSheet (3) Data (1) }
```

Comment peut-on lire des choses pareilles quand tout montre le contraire !

Alors des noms ou des numéros, espace plat ou autre ? Pourquoi vous excitez vous sur des numéros ? C'est ridicule ! J'en sais rien, pourquoi l'informatique y échapperait ? Après tout, rien n'a jamais empêché tout autre espace de références de publications d'utiliser des noms et non des numéros, dynamique, capacité de consommation d'identifiants pour pouvoir en changer ou conserver d'autres, ceci dépasse le problème des trademarks, il y a tous les identifiants que l'on pourrait qualifier de type ou noms communs, par essence partagés.

Tout d'abord: beaucoup d'espaces dits hiérarchiques utilisant des noms n'ont en fait rien de hiérarchique, mais représentent des compositions de symboles, "X11/bin" ou "X11/man" représentent les connexions "X11—bin" ou "X11—man", X11, bin et man étant en fait au même niveau, c'est à dire, par rapport aux Object-ID, à la composition de deux Object-ID, (reste alors à savoir si l'on veut donner le même sens, (identifier la même écriture) à travers "X11—bin", et "bin—X11", question de choix). Ça n'est pas parce que l'on manipule des listes qu'elles correspondent à des espaces hiérarchiques ! Pour en revenir à l'exemple précédent, MS-DOS-Application ou Spreadsheet pourraient être des Object-ID en eux-mêmes susceptibles d'être utilisé par la compagnis PC-Software-bis. C'est bien là le problème, dans le cas d'espaces de noms, là où on croit manipuler des espaces hiérarchiques, on batit en fait un modèle de données qui n'a lui rien de hiérarchique par l'intermédiaire d'un top level qui n'est pas considéré en tant que tel. Alors que dans le cas des numéros (Object-ID) , ces espaces sont effectivement considérés de manière hiérarchique. En résonnant en terme de sources de connecteurs, il n'y en a qu'une dans un cas (partie référence de la hiérarchie UNIX ou similaires), une par nœud dans l'autre. Dans la partie référence d'unix, la chaîne "/a/b/c" peut être interprétée comme valeur de c dans le contexte b dans le contexte a, et la chaîne /a/d/c" comme valeur de c dans le contexte d dans le contexte a. Ceci n'a rien à voir avec une hiérarchie, le côté hiérarchique vient lui de l'écriture. Cependant, la hiérarchie UNIX peut aussi parfois être considérée comme hiérarchique, on a toujours le choix entre — tenir compte — du fait qu'il y a des identifiants égaux dans différentes branches ou pas. Si l'on en vient aux Object-ID, la même chose peut aussi se passer: n1/n2/n3, et dire, les n3 sont

[7]Mais le fait qu'un livre ait un numéro ISBN ne l'empêche pas d'avoir également un numéro plus petit dans la collection dont il fait parti. L'interface de nommage du livre est plus simple structurellement vers l'extérieur (numéro ISBN), plus compliqué structurellement mais l'identifiant moins long à l'intérieur (dans la collection de l'éditeur). Et puis, à l'heure actuelle, beaucoup de Booléen sont écrits sur 8,16, ou 32 bits.

les identifiants définis dans la norme xxx. Cependant le problème est que l'on ne peut espérer réaliser ce qui se fait dans la hiérarchie UNIX vue comme modèle de données car, de fait on ne considère pas une source d'identifiants unique, on ne parle pas de la même chose, considérer de manière plate les numéros introduits dans les diverses branches de la hiérarchie Object-ID n'a de toute évidence aucun sens. Vous me direz, pas grave, bâtissez une hiérarchie style UNIX où les noms sont des Object-ID (des listes), et le tour est joué, car en tant que liste les Object-ID constituent une source de connecteurs unique. Sans doute, mais il reste le fait que la hiérarchie Object-ID existe, et:

- Il semble que l'on veuille parfois lui faire réaliser ce que fait une «hiérarchie» comme celle d'unix, confusion des rôles, les numéros ne sont plus du tout utilisables (pas de la même manière), l'information introduite par les noms étant supérieure par utilisation de la combinatoire sur un espace plat non géré. Et si les noms sont "parallèles" aux numéros, à quoi bon des numéros ? L'intérêt des numéros ne devrait il pas être de pouvoir créer des homonymes ?
- Les problèmes de tailles et d'atomicité arrive un jour ou l'autre (une écriture n'est jamais fractale), si l'on utilise des numéros, au moins qu'ils soient souvent utilisables, dans le cas contraire, d'autres espaces apparaissent rajoutant quelques vrais synonimes, et le premier espace n'est plus utilisé n'étant plus nécessaire.
- Problème de l'accés aux identifiants pour limiter le recours aux extrémités du type:

```
/*  This file is (C) 1994 David W. Flater. */

/*  Choose port numbers for Ubernet daemons.  A daemon must bind to one of
 *  these ports when it initiates a link to pass authentication.  The method
 *  of choosing port numbers is admittedly quite silly.
 [...]
#include "alibi.h"
 [...]
/* This function estimates how "photogenic" an integer is, i.e. how likely
 * it is that somebody else will want to use this number as the port number
 * for _their_ program.  Decimal and hexidecimal bases are considered.  There
 * are 472 completely nonphotogenic port numbers.
 */
```

Reprenons: nécessité de constitution d'espaces plats, ou plutôt de sources uniques, quelques niveaux organisationnels dans un espace de numéros n'est pas un drame. Limitons nous à un seul. D'une certaine manière, quand on utilise des noms, ces espaces plats sont sous entendus, n'étant pas gérés en tant que tels et sont donc par là-même ouverts de fait. Quand ils sont gérés, ils sont plus ou moins ouverts. Mais, dans le cas des noms non gérés (mots clés), le problème se reporte souvent sur les index qui sont plus ou moins assumés, du fait que l'unicité n'est pas forcément considérée à priori, exemple :

```
You still have to tell the Ada (or any other) compilation system (and by
that I mean each part of it) where to find it's modules, don't you? What
about links to foreign languages? What happens if two modules you've
acquired have the same name?

Whether it's done by compiler switches or environmental variables, the
machine dependencies have to be built in at some level or other sooner
or later.
```

Ceci est de toute évidence incompréhensible (les machines sont aussi des livres), maladie d'Unix (ou plutôt de la manière dont le modèle est étendu). Mais aussi problème difficile, car les index (de références :/usr/bin /lib..., etc ...) étant les identifiants les plus stables, sont aussi les plus durs à faire bouger (ayant le plus de connexions[8]), et on a toujours le choix entre, par exemple, pour un langage L, créer un index /lib-L ou utiliser /lib/L (notons que ceci implique un consensus plus large entre les divers concepteurs de langages). Mais c'est à mon avis là que la dynamique doit être garantie à travers un top level de numéros (éventuellement branché par un nom pour ne pas effrayer certains), où n'importe qui peut venir s'accrocher, en choisissant ses formes d'indirections. Quant aux machine dependencies, ceci peut toujours se résumer à installation/désinstallation (si l'on considère un

[8]J'ai lu à ce sujet que la structure des index actuellement utilisée sous UNIX: /bin, /usr/bin, /local/bin découlait de certaines limitation mémoire des premières machines utilisées.

module name comme ce que c'est, c'est à dire une référence ubiquitaire, il peut y avoir plusieurs modules désinstallés ayant le même nom).

Il y a aussi bien sur le cas espaces de noms gérés de manière effective, par exemple l'espace de noms des formats associé à la norme Internet MIME. A ce sujet, on peut se demander pourquoi une classification a encore été introduite selon les lignes son, image, données (ici `application` (rôle de `misc`)), quand ces messages sont justement sensés être multimédia.

Les scissions d'espaces de noms sont typologiques ou organisationnelles. Typologiques: pourquoi mélanger deux "choses" que l'on n'aura jamais besoin de comparer, (à travers leurs identifiants), mais en oubliant que pouvoir les considérer dans un même index peut être en soi nécessaire (ou utile), et que de toute manière, ça ne retire rien. Organisationnelle: il est difficile de partager un même distributeur de numéros, sans en venir au caractère révulsant. La scission typologique correspond à un réel choix, mais les types évoluent, ils leur arrivent de se simplifier, si elle est souhaitable avec des noms (permettant de créer des homonymes), elle est ridicule quand il s'agit de numéros (sous certaines limites, on pourrait aussi mettre la lettre A à côté des livres), ce qui n'empêche pas après de considérer des types (ensemble). La scission organisationnelle est elle plus difficilement évitable, mais les espaces de numéros hiérarchiques passant à l'échelle sont des mirages (car les identifiants les constituant ne sont jamais innocents: toujours sujet à consensus éventuel). Vouloir trouver un modèle d'évolution d'écriture, en dehors de toute forme, ce qui est le cas, n'a pas de sens (ceci est d'ailleurs vrai dans une certaine forme, une modification d'écriture est toujours une révolution). Cependant, si numéros, on peut avancer la règle que les structures d'étiquettes reflétant les composantes organisationnelles doivent être réduites au maximum, celles reflétant la typologie évitées, la manière dont sont utilisés les Object-ID est un scandale! Car bien qu'un espace hiérarchique puissent être considéré comme source d'identifiants uniques, les seules "vraies" sources (celle qui se font sentir à un moment ou un autre), sont aux nœuds de cet espace, et ce sont celles qui entre en jeux dans tout processus de modification des structures.

Ajoutons que de tels espaces n'attendraient pas d'être utilisés par les systèmes pour être utile à leur évolution (si ouverts aux concepts existants), à ce sujet, le World Wide Web est actuellement en recherche d'un tel espace (draft-ietf-uri-resource-names-02.txt), qui s'appellerait alors URN (uniform resource name), au lieu de URL (uniform resource locator), et qui, au niveau écriture, serait plus plat que le précédent:

```
URNs are not required to be human-readable in the sense that a human could
look at the URN and determine anything about the contents of the resource.
While the Naming Authority (q.v.) has the final determination of the contents
(subject to the syntax constraints), the Naming Authority is STRONGLY
discouraged from placing metainformation about the resource into the resource's
URN, as the URNs are not expected to be read, and because this paper will
specify only five consistent components of the URN. Although there have been a
number of proposals placing extensive semantics on the contents of the URN
[Spero 1992, Kunze 1993], it was decided by the authors of all the proposals
that all metainformation should be conveyed using another mechanism, and that
the Naming Authority should assume that humans will never look at the contents
of the URN to determine qualities of the resource they are retrieving, and
would not be required to guess from a given URN the URN of a document which
might be related.
```

Il y a toujours cette histoire de human readable ou pas, on entend: «les numéros sont pour les machines, les noms pour les humains», alors que les mots clés conviennent aux machines tout comme les numéros (parfois des contraintes de remplissage d'espace, d'utilisation, ce qui n'empêche pas d'autres identifiants dans un contexte englobant), manière d'éviter la pointe, et le fait que l'on — veut— conserver ou faire évoluer certains identifiants textuels, les numéros sont pour la dynamique des mots clés, de l'écriture et des listes, j'exige des numéros !

En résumé:
Les espaces de noms dits hiérarchiques ne sont pas hiérarchiques. La partie référence d'UNIX peut être considérés comme un langage dont l'alphabet évolutif est constitué des symboles "bin", "lib",

"README" ..., et les chaînes valides, les chaînes utilisées à l'heure actuelle. Ceci permet de créer des identifiants sans augmenter l'alphabet, ce qui est essentiel, et de constituer une grammaire.

Ce schéma n'est pas celui d'une hiérarchie comme celle des Object-ID qui est elle vraiment hiérarchique, créant une source d'identifiant (un alphabet) à chaque nœud de la hiérarchie.

Des contraintes administratives ou disont de partage ou nécessité de champ de bataille et d'évolution des identifiants textuels aboutissent à une certaine lourdeur ou sclérose dans le cas des espaces de noms et puis l'horreur de la ligne, associée à une non considération ou connaissance du premier point.

Créer l'équivalent de ce qui se fait dans le cas des espaces de noms hiérarchiques en introduisant des numéros implique d'applatir la stucture de ces numéros pour ne pas trop mélanger les deux notions (langage ou ensemble de chaînes sur un alphabet / hiérarchie).

Il semblerait que l'utilisation actuelle du concept d'Object-ID soit un scandale. On voudrait ici bénéficier de l'utilisation de numéros, c'est à dire d'avoir par exemple la possibilité pour deux acteurs d'accrocher au même point, deux choses considérées avec le même nom à travers deux tables de correspondances nom-numéros différentes, mais en même temps, éviter le pendant essentiel de cette approche qui est de limiter au maximum le nombre de sources de numéros (ou qu'il en existe certaines franchement ouvertes, ce qui revient au même), les numéros sont supportables en proportion inverse du nombre de leurs sources, et, si il est possible de changer de root en gardant la même source, l'inverse n'a pas de sens.

En d'autres termes, le concept d'Object-ID n'est rien, on peut avec n'importe quels espaces d'identifiants existants, rajouter un niveau pour créer un espace englobant, problème classique du «black penny syndrome», (le fait que la poste Anglaise ayant inventé le timbre poste n'ait pas jugé nécessaire d'y inscrire le nom du royaume (dicussions à ce sujet sur `comp.dcom.telecom`), ou «un ou plusieurs, cela ne saurait revenir au même» (Sophocle)), tout bénéfice vient de la constitution d'axes (de sources), et dans le contexte informatique, en ouvrir un pour les langages, formats, librairies, modems, claviers ou espaces existants, n'a rien de délirant. Après tout, les colored books, documents définissant les normes du mondes ISO/CCITT ont un numéro ISBN (numérotation qui est également une norme ISO), et sont publié — à côté — d'Hélène et les garçons en Iliade ou romans, cartes Michelins, ou plus de 150 bouquins publiés au sujet de C++ (il y a 2 ans).

Et aussi, quelle est donc cette sagesse qui dirait: «on sait pas si, on sait pas ça, en attendant on ne fait pas comme on a toujours fait», surtout face à des problèmes de toute évidence insolubles, ou — plus précisément —, Absents (il n'y a que des solutions).

Qui a droit aux numéros ?
(qui les donne)

Nous touchons là un point sensible du vertige d'Alice de Lewis Carrol:
«"The question is", said Alice, "wether you *can* make words mean so many different things."
«"The question is," said Humpty Dumpty, "which is to be master—that's all."»
Pas d'amalgame, nous parlons d'identifiants de concepts de représentations, de concepts de création, acquisition, consultation, transformation, échange, stockage de ces représentations, mais pas des mots ou signes que ces représentations peuvent véhiculer. De mots clés dont l'utilisation, la ventilation, ne suffit pas à modifier le sens (ou ensemble de) qui lui n'évolue qu'au rythme des publications écrites modifiant le système, d'un domaine ou le fond n'a jamais fait bouger la forme. De mots clés dont la ventilation ne dit rien puisqu' elle a lieu dans un système parfait (écrit), où la séparation utilisation d'un mot/modification d'un mot est toujours on ne peut plus claire.

Donc, si on peut classifier les logiciels (concepts télématiques, du clavier au compilateur), c'est sans doute d'abord en séparant les logiciels publiables (dont la publication a un sens, un marché) de ceux qui ne le sont pas. Si le nombre de logiciels non publiables est plus important que ceux qui le sont, les logiciels non publiables utilisent de plus en plus de logiciels publiés, tant pour leur écriture que pour leur utilisation. La «conceptographie» de tout logiciel augmente, ceci montrant également que la réutilisation devient une réalité, les écritures de ces conceptographies sont actuellement en recherche de colonnes vertébrales, il semblerait qu'il en faille beaucoup (de vertèbres, pas de colonnes). De plus, en reprenant les deux sens d'ouvertures précédemment définis, dans la pratique, c'est toujours le système d'exploitation (son espace de référence) qui a le dernier mot, mais pour que cela change, il est nécessaire de pouvoir référencer dans les conceptographies n'importe quel concept disponible (ouverture des espace existants).

Conclusion

Si le langage (capacité au, d'évolution ou de nettoyage), Azur épris de synchronisme, cité interdite à jamais non inscriptible (ou toujours écrite), ivresses assassines, et les ventilateurs, portée ou traînée par une multitude, il s'agit ici d'une multitude moins grande batissant *une* écriture, hommage à la nature, un peu comme la littérature.

Plusdans-unpeucomme, non pas pour mettre du continu ici immoral, mais l'informatique et les télécoms, la technique, écrivent une immense encyclopédie, que l'on peut voir comme un texte ou comme un graphe (sans notion de dimension), où chacun y va de son paragraphe (contrairement à la littératures, cinéma, jeux électroniques, ou enregistrements télévisés). Il suffit de le dire, de le désigner, quant à savoir si tout est lié de manière transitive, si vous trouvez une île, vous pouvez écrire entre les deux. A la différence d'une autre encyclopédie, celle-ci est intrinsèquement parfaite (au sens moderne) puisque tautologique, faite de raison pure (même si baroque, du fait écrit, où toute connexion utilisable est élucidée ou élucidable) ainsi que d'adéquation (création découverte) au besoin, rarement lue (directement) mais utilisée en permanence, transparente. Elle contient des définitions de formes de publications, modes d'interactions, espaces de références d'autres écrits ou enregistrements toujours morts à durée de vie (lecture) variable, imposant à ces écrits une composante statique tant dans leurs formes que structures d'échanges (règles de visibilité), où lois et directives administratives viennent aussi s'écrire.

De ce caractère tautologique, cette écriture utilise non seulement des normes ou produits mais également leurs noms, la structure de ces noms fait donc intégralement partie du système. Parler de grammaire, syntaxe, modèle, alphabet, root, à la surface, n'a pas de sens, il n'y a qu'une écriture ouverte à toute nouvelle syntaxe et, *point important*, rien n'a jamais empêché toute nouvelle écriture dans toute nouvelle syntaxe de se connecter par l'intermédiaire d'identifiants précédemment définis (connectés) dans d'autres syntaxes, ce qui se passe en permanence.

La structure de noms actuellement utilisée est constituée de divers espaces (axes) certains considérés comme tels d'autres plus ou moins. Il est permis de se demander si cette structure ne pose pas actuellement autant de problèmes d'évolution que la complexité intrinsèque, tant dans la sphère documentaire que dans celle des définitions effectives.

Le problème est avant tout politique ou organisationnel, face au nombre d'identifiants nécessaires, n'y a-t-il pas pour réduire les structures (ou constituer de vrais axes), nécessité de vrai séparation entre organismes numéroteurs de concepts et organismes concepteurs de concepts, je veux dire de distribution par paquets et non par structures, et d'oublier la frontière normes/produits ?

Annexe

Définition:

«
A “symbol” is an abstract entity that we shall not define formally, just as “point” and “line” are not defined in geometry. Letters and digits are examples of frequently used symbols.
»
(«Introduction to automata theory, languages, and computation», John E. Hopcroft, Jeffrey D. Ullman)

On pourrait sans doute se révolter contre cette définition, Ecrire ça ! Vous voulez rire ! et la modernité ?!:
« Notes.
Il a été démontré par la lettre — l'équivalent de la Fiction, et l'inanité de l'adaptation à l'Absolu de la Fiction d'un objet qui en ferait une convention absolue.
» Mallarmé.

Et aussi faire attention:
«Des faibles se mettraient *à penser* sur la première lettre de l'alphabet, qui pourraient vite ruer dans la folie !» (Rimbaud)

Remarquons que dans la définition ci-dessus, le mot “symbol” désigne en fait un ensemble, alors que dans le cas ligne ou point, il n’y a qu’une ligne et qu’un point au sens “abstract entity”, ce qui nous amène au fini/infini ainsi que problème de niveaux dans les ensembles. Les langages formels sont il est vrai en général basé sur un alphabet fini (ceci fait d'ailleurs souvent parti de la définition d'un alphabet), mais mathématiquement, il n'y a qu'un alphabet à n éléments, ou, si on en considère deux, n! possibilités de correspondances, cependant il faut écrire, communiquer ..., d'où les symboles, entre les deux.

Il ne s'agit pas non plus de dire que la théorie des automates ou des langages formels ne sont pas des mathématiques (ni des méta mathématiques), alors quoi ?

Reprenons, programme de Hilbert, Russel et Withehead, Principia Mathematica : les écritures mathématiques doivent aussi être des objets mathématiques, ou manière d'approcher le problème (ou des représentations isomorphes, il reste toujours le côté matière de toute écriture, ou plutôt, sans en venir à la matière[9], au choix des symboles, une image accédant au statut de symbole quand elle est considérée comme tel), puis, — Gödel —, et aussi Wittgenstein:
«(Le «signe de l'assertion» "⊢ " de Frege est logiquement sans signification; chez Frege (et chez Russel) il indique seulement que ces auteurs tiennent pour vraies les propositions ainsi désignées. De ce fait "⊢ " n'appartient pas plus à la structure de la proposition que le numéro de la proposition. Il est impossible pour une proposition d'affirmer d'elle même qu'elle est vraie.)»
Qu’en est il des machines (programmes ou formats), en nombre infini de toute évidence mathématique ? (tout comme les théorèmes mathématiques). Car, si parler de la vérité d'une machine ou d'un programme n'a pas de sens (bien qu'un discours de vérité mathématique soit possible dans le contexte de la machine), il paraît difficile de nier leur «existences» effectives ou capacité à relier le monde de la connaissance à celui des phénomènes[10]. Quoiqu'il en soit, si il y a un mot sur utilisé en informatique, c'est bien le mot «abstrait» là ou il n'y a jamais que volonté d'échapper au couteau de l'écriture (ou parole) et à la multitude potentielle de ces écritures, par exemple dans les expressions

[9]«Si vous dites qu'une phrase parlée est un évenement physique composé de certains mouvements de la matière et qu'une phrase écrite se compose de signes d'une certaine couleur sur un fond d'une autre couleur, on vous trouvera vulgaire.» Russel

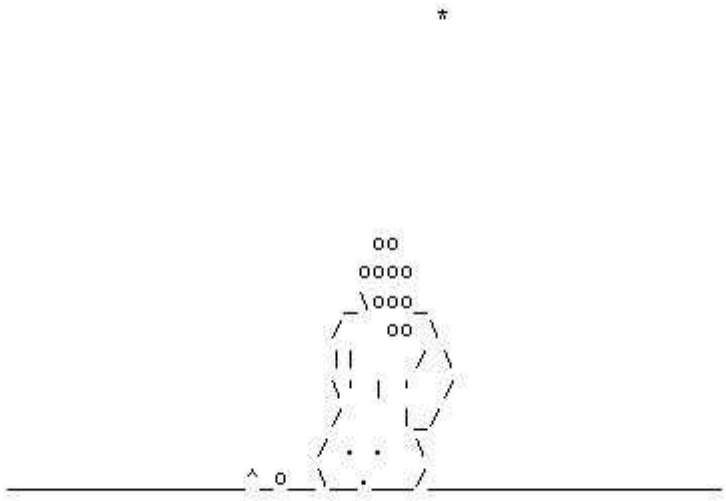
[10] De plus, le fait que les machines (ou programmes) sont des écritures mathématiques elles mêmes objets mathématiques, oblige à ne pouvoir parler que de *la* machine, de considérer de manière unique ce qui dans le temps peut toujours se connecter.

"abstract data types" (comme si il y en avait de plus abstraits que d'autres, pourquoi préciser que l'on fait des mathématiques quand l'on en fait ?), souvent associé à cette espèce de croyance à une différence fondamentale entre hardware et software, matériel logiciel, je ne vois pas en quoi la question "combien de caractères contient la page 5 du livre 2345678912 ?" est plus ou moins scientifique ou technique que la question "combien de broches a le microprocesseur XYZ234 ?", la question du coût de production (au sens faire des copies) reste il est vrai fondamentale. Cette manie de considérer le logiciel comme quelque chose d'immatériel (ce qui est permanent à l'heure actuelle, économie de l'immatériel et compagnie) est fausse tout d'abord parce qu'un logiciel nécessite toujours un support matériel, mais surtout, une voiture ou aspirateur contiennent autant de ce que l'on veut bien appeler immatériel dans le cas du logiciel (production de l'esprit, du sens, de la connaissance, information), ou «Elle est en marbre ou quoi la Venus de Milo ?» .

Il y a aussi : «on va localiser les lettres ou mots dans une cervelle, puis implants cérébraux, interaction par la pensée avec une machine...» (pub récente à ce sujet: les savants du MIT pensent qu'avant dix ans ...) Je n'ai de toute évidence pas — Le Temps — de montrer la bêtise de telles approches sans même tenter d'en comprendre les desseins, enfin bref ...

Postface

Conclusion, amalgame, mélange des genres, ne pouvant être que directement politique, ce qui n'arrange rien à l'affaire. Cependant, un livre s'écrit, c'est tout ce qu'on peut en dire, si ce n'est que ses capacités évolutives nécessitent un certain gaz d'éclairage. La solution, souvent utilisée, effleurée ou réclamée, est toute trouvée. Quoiqu'il en soit, il y a un problème, c'est clair, qu'on en parle, il y a à mon avis : Urgence



Là-bas, dans leur vaste chantier
Au soleil des Hespérides,
Déjà s'agitent — en bras de chemises —
Les Charpentiers.

Dans leurs Déserts de mousse, tranquilles,
Ils préparent les lambris précieux
Où la ville
Peindra de faux cieux.

Ô, pour ces Ouvriers charmants
Sujets d'un roi de Babylone,
Vénus ! quitte un instant les Amants
Dont l'âme est en couronne.

Ô Reine des Bergers,
Porte aux travailleurs l'eau-de-vie,
Que leurs forces soient en paix
En attendant le bain dans la mer à midi

Rimbaud, « Une saison en enfer », alchimie du verbe