Zeitschrift: Technische Mitteilungen / Schweizerische Post-, Telefon- und Telegrafenbetriebe = Bulletin technique / Entreprise des postes, téléphones et télégraphes suisses = Bollettino tecnico / Azienda delle poste, dei telefoni e dei telegrafi svizzeri

Herausgeber: Schweizerische Post-, Telefon- und Telegrafenbetriebe

Band: 67 (1989)

Heft: 4

Artikel: Le breuvage du verre à vin OSI

Autor: Pitteloud, Joseph

DOI: https://doi.org/10.5169/seals-874929

Le breuvage du verre à vin OSI

Joseph PITTELOUD, Berne

Couches de présentation et d'application et enjeux

Das Getränk im OSI-Weinglas – Präsentations- und Anwendungsschichten sowie Zielsetzungen
Zusammenfassung. *Der Autor gibt einen Überblick über den Ende 1988 erreichten Normierungszustand der Präsentations- und Anwendungsschichten in der OSI-Architektur und versucht die Zielsetzungen und die Anwendbarkeit der Normen kurz zu erläutern. Einige Entscheidungselemente sind dargestellt.*

Résumé. *L'auteur aborde de manière synthétique l'état, à fin 1988, de la normalisation des couches de présentation et d'application dans l'architecture OSI et tente d'esquisser les enjeux d'OSI et l'applicabilité des normes. Quelques éléments de décision sont proposés.*

Gli strati OSI di presentazione e applicazione – obiettivi
Riassunto. *L'autore presenta una sintesi dello stato di normazione – alla fine del 1988 – degli strati di presentazione e applicazione nell'architettura OSI e illustra brevemente gli obiettivi e l'applicabilità delle norme. Fornisce inoltre alcuni elementi decisionali.*

1 Généralités

Dans un article précédent («Bull. Techn. PTT» No 1/89) le tableau synthétique de l'architecture OSI (Open System Interconnection) a été brossé. La couche de transport et celle de session ont été également explicitées. Dans cet article, l'auteur aborde les deux dernières couches de l'architecture (présentation et application) ainsi que les enjeux d'OSI. Certaines comparaisons ayant été faites dans le premier article, sa lecture est recommandée.

2 Etage de la présentation

21 Les syntaxes

211 La notion de syntaxe

La syntaxe définit la représentation binaire des éléments de données à échanger. Elle est le support de la sémantique qui définit la signification conceptuelle spécifique pour une application des différentes données. On distingue quatre syntaxes *(fig. 1* et *2):*

- une *syntaxe abstraite,* qui définit l'application de manière indépendante d'un langage de programmation. C'est en quelque sorte une langue conceptuelle, une méta-langue au niveau du *signifié* en linguistique. Elle permet de décrire le sens des informations indépendamment du support. Ce sens est la sémantique comprise et partagée par les deux officiers
- une *syntaxe concrète du système ouvert A,* qui est celle du langage de programmation utilisé (Pascal, par exemple, la langue française de l'officier)
- une *syntaxe concrète du système ouvert B,* qui est, dans la plupart des cas, différente de celle utilisée par A (par exemple langage Unix C, la langue allemande de l'autre officier)
- une *syntaxe concrète de transfert,* qui définit les règles communes d'échange entre les systèmes, c'est-à-dire les règles de codage et de décodage que doivent exécuter chacun des systèmes ouverts, pour pouvoir localement utiliser leur propre syntaxe, tout en étant capables de communiquer, pour l'application donnée, avec le système opposé. Cette syntaxe définit la valeur des octets «user data» de la couche session qui transporte les valeurs de la couche application (la langue anglaise de l'analogie).

Les trois syntaxes concrètes sont du niveau du *signifiant* en linguistique, c'est-à-dire la manifestation matérielle du signe, la suite de phonèmes ou de caractères qui constitue le support d'un sens.

Actuellement, une seule syntaxe abstraite est définie (X.208) et une seule syntaxe de transfert (X.209). On ose espérer que les normateurs vont s'arrêter là, sinon il n'y a que peu d'espoir de voir se créer un *anglais esperanto* syntaxique entre systèmes ouverts.

212 La syntaxe abstraite (CCITT X.208)

La syntaxe abstraite appelée ASN.1 (Abstract Syntax Notation One) est une notation formelle utilisée pour décrire les applications de manière indépendante de la représentation binaire concrète des éléments de données à transmettre appelés «types».

Cette notation utilise les mêmes principes de définition que ceux d'un langage de programmation. Chaque type de données est identifié par un nom réservé de classes

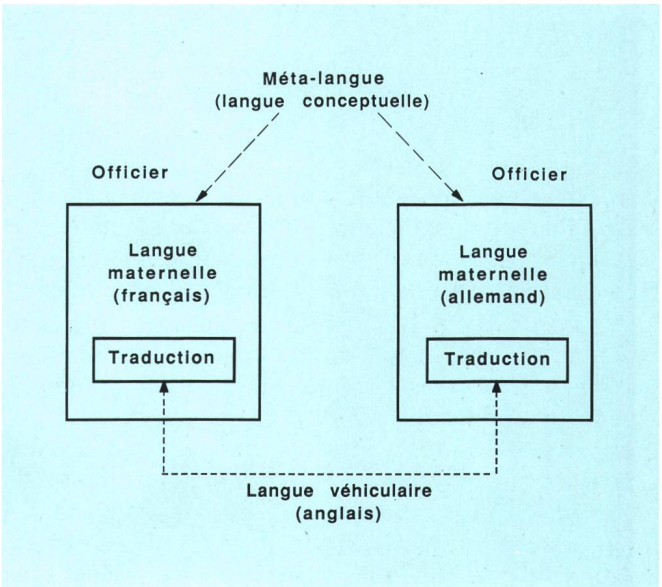

Fig. 1
Langues

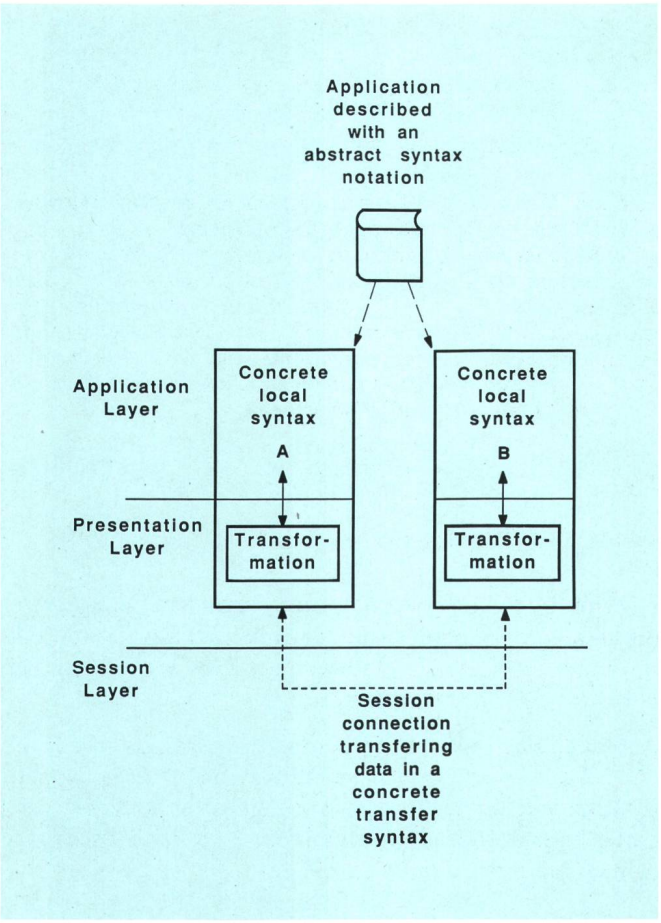

Fig. 2
Syntaxes

(tag class) et son numéro dans la classe donnée (tag number). La notation permet également de spécifier des valeurs à ces types. On distingue quatre classes:

- *classe universelle:* Utilisable dans toutes les applications, par exemple:
  - des types simples, tels que BOOLEAN, INTEGER, BITSTRING, OCTET, STRING, etc.
  - des types définisssant des chaînes de caractères spécifiés par des jeux de caractères normés, tels IA5STRING, TELETEXSTRING, VIDEOTEXSTRING, etc.
  - des types qui sont construits à partir de types simples, tels une suite ordonnée (SEQUENCE), une suite non ordonnée (SET), un choix (CHOICE) entre types.
- *classe valable pour toute l'application:* Ces types sont spécifiques aux applications. Chaque application peut assigner ainsi des «tags» propres, qui ont un sens bien précis pour les deux systèmes ouverts: Ainsi le «tag» [APPLICATION 0] pour une application bancaire déterminée sera le numéro de compte, tandis que le «tag» [APPLICATION 0] pour une application messagerie sera une adresse de destinataire.
- *classe dépendant du contexte* (context specific): Ces types sont des classes qui permettent, dans le contexte concret d'une application, de distinguer un type d'un autre. Ainsi lorsque, dans l'application messagerie, l'en-tête du message est transmis, c'est avec ce type de «tags» que l'on distingue les différents éléments dans la séquence de l'en-tête, tels que la référence du message auquel il est répondu, le sujet du message, etc.
- *classe privée:* Ces types sont des classes prévues pour des applications intra-entreprises, non normées par l'ISO ou le CCITT.

Vu le caractère récursif de la notation, il est possible de définir des structures arborescentes de données très complexes que s'échangent les applications. Typiquement, dans ces structures de description abstraite des données échangées, on trouve les types de classe «valable pour toute l'application» dans le tronc et les branches principales *(fig. 3)* (A[0] et A[2]). Les embranchements à partir du tronc, des branches principales jusqu'aux branchilles, sont formés par des types universels construits (SET, SEQUENCE, CHOICE). Les types spécifiques au contexte servent à identifier, dans un SET par exemple, les différentes branches (C[0], C[1]). Quant aux contenus sémantiques, ils sont portés par les feuilles, définis par des types universels primitifs (NUMERICSTRING, IA5STRING, etc.).

213 La syntaxe de transfert (CCITT X.209)

Une seule syntaxe concrète de transfert est définie aujourd'hui pour la syntaxe abstraite ASN.1. Chaque valeur de données est codée et transmise sous la forme illustrée par la *figure 4:*

- 1 ou plusieurs octets d'identification
- 1 ou plusieurs octets de longueur
- 1 ou plusieurs octets de contenu.

L'identification définit la classe du type de données (tag class ASN.1) et, dans cette classe, le numéro du type en question. Ainsi, dans une application bancaire, si [AP-

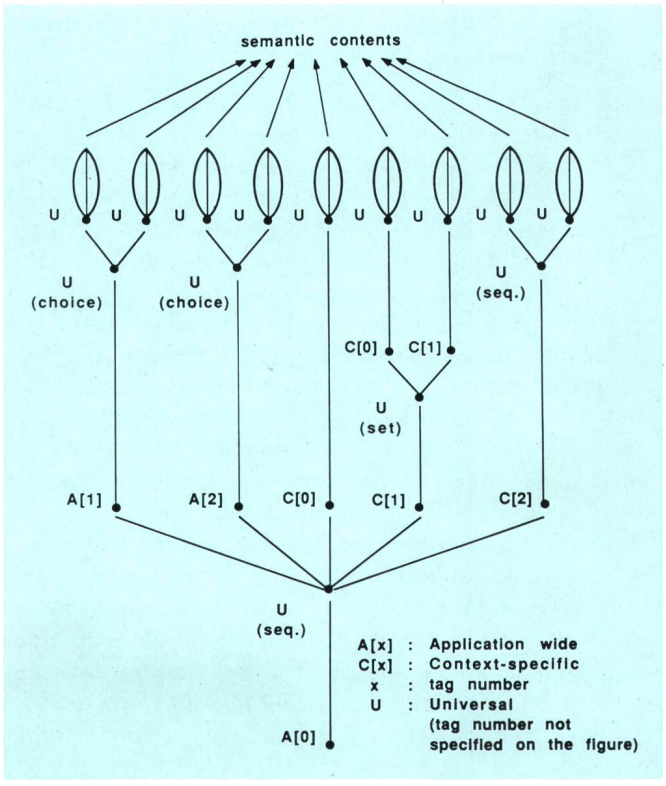

Fig. 3
Structure sémantique en arbre représentée par une syntaxe abstraite

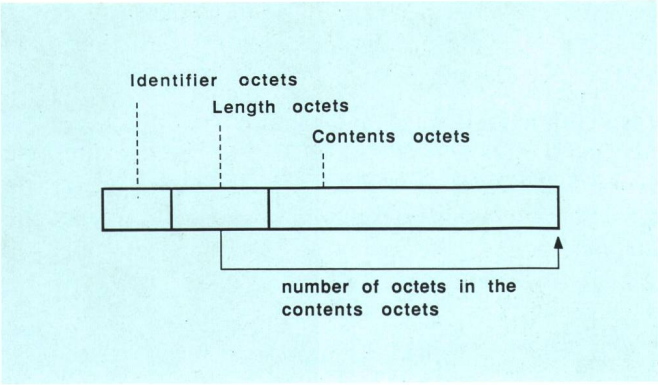

Fig. 4
Structure de codage de la syntaxe de transfert

PLICATION 0] est le numéro de compte, [APPLICATION 1] peut être le nom du propriétaire du compte. Dans le cas d'un seul octet d'identification, les bits huit et sept déterminent la classe (UNIVERSAL, APPLICATION, CONTEXT-SPECIFIC, PRIVATE) et les bits cinq à un déterminent le numéro dans la classe. Le bit six détermine si le codage est un codage primitif (primitive encoding) ou un codage de construction (constructed encoding). Dans le cas du codage primitif, la valeur du contenu a une signification pour l'application (feuille de l'arbre); dans l'autre cas, le contenu est une construction récursive qu'il faut disséquer en éléments constitutifs pour trouver, en bout de chaîne, des types primitifs (tronc et branches de l'arbre, fig. 3 et *5)*.

La longueur définit le nombre en octets du contenu. Il existe trois manières de la spécifier:

- la forme courte qui permet de spécifier des contenus jusqu'à 127 octets
- la forme longue qui utilise plus d'un octet pour spécifier la longueur
- la forme indéfinie qui permet, par un champ spécial de fin de contenu (tag de classe universelle), de définir quand le champ du contenu est terminé.

Le contenu est la substance à transmettre. Si le champ est de la forme primitive, la valeur du contenu peut être interprétée sémantiquement par l'application. Sinon, le système ouvert devra décortiquer le contenu pour trouver la sémantique, c'est-à-dire les valeurs primitives contenues dans les feuilles de la structure arborescente (fig. 3).

22 Le service et le protocole de présentation (CCITT X.216, X.226)

221 Le rôle de la couche de présentation

La couche de présentation, contrairement à ce que son nom pourrait laisser supposer, n'est pas destinée à la présentation des données (à un terminal, par exemple), mais à la *représentation* des informations entre deux applications de deux systèmes ouverts. Pour que deux entités d'application (officiers) soient à même de communiquer, il faut non seulement qu'elles définissent l'application qu'elles veulent traiter (rôle spécifique) mais encore qu'elles négocient, pour cette application, la (ou les) syntaxe(s) d'application utilisée(s) (méta-langue). La couche d'application (processus mental supérieur) en informe la couche de présentation (processus mental inférieur) qui doit ensuite négocier avec le partenaire de l'autre système quelles sont les syntaxes de transfert acceptables mutuellement (langue anglaise par exemple) pour ces syntaxes abstraites imposées. Ensuite, pendant toute la durée de l'association entre les applications, la couche de présentation doit réaliser les codages et décodages nécessaires entre la syntaxe de transfert négociée et la syntaxe concrète locale du système ouvert (langues française et allemande de notre exemple, fig. 1 et 2).

L'association entre une syntaxe abstraite imposée par l'application et une syntaxe de transfert négociée par la présentation s'appelle un contexte de présentation, ou *«presentation context»*.

222 Les mécanismes de présentation

Trois mécanismes ont été prévus pour la définition de contextes de présentation *(fig. 6)*:

- *L'agrément antérieur (prior agreement)*
  Dans ce cas, les valeurs du contexte de présentation sont connues des entités d'application des deux systèmes ouverts avant même l'ouverture de la connexion de présentation par convention bilatérale antérieure.
- *Le contexte par défaut (default Context)*
  Le contexte par défaut est défini à l'établissement de la connexion de présentation puis est utilisé durant toute la durée de la connexion, sauf si le mécanisme suivant est mis en place.
- *La collection de contextes prédéfinis (Defined Context Set, DCS)*
  Dans le cas d'une collection de contextes prédéfinis, (DCS), chaque système possède une liste des contextes de présentation négociés. Chaque contexte ainsi négocié est disponible pour utilisation immédiate par l'application (l'anglais, l'espagnol ou le russe dans notre exemple).
  Cette liste DCS est négociée en début de connexion de présentation et peut être modifiée en cours de connexion (alter-context service), c'est-à-dire qu'il est possible d'ajouter ou de supprimer des contextes de

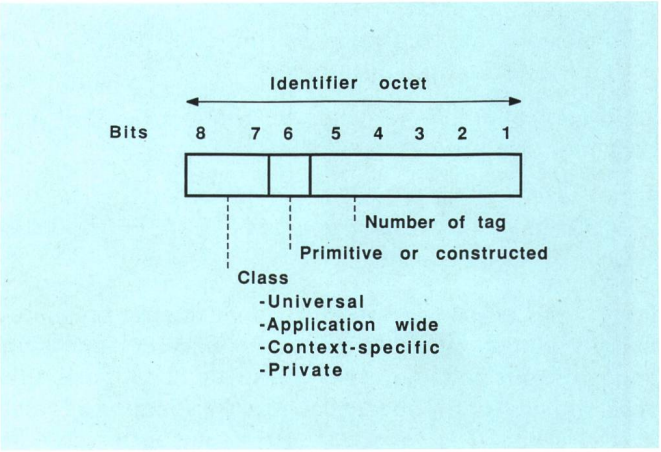

Fig. 5
Octet d'identification isolé

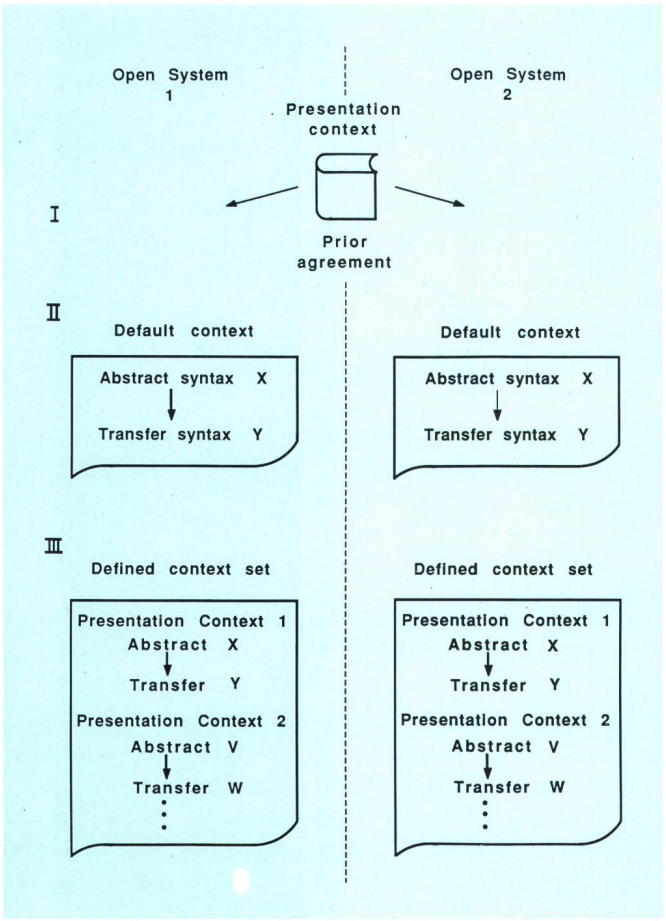

Fig. 6
Mécanismes de présentation

présentation dans la liste DCS. Il est donc supposé que les officiers ont un don des langues assez remarquable!

223 Les principes du protocole

L'établissement du contexte par défaut et de la liste de contexte DCS initiale se fait par l'échange des unités de données du protocole de présentation (Presentation Protocol Data Units PPDU):

- connexion de présentation CP PPDU (Connect Presentation)
- connexion de présentation acceptée CPA PPDU (Connect Presentation Accept)
- connexion de présentation refusée CPR PPDU (Connect Presentation Reject).

La modification de la liste de contexte se fait par l'échange des protocoles suivants:

- contexte modifié AC PPDU (Alter Context)
- confirmation du contexte modifié ACA PPDU (Alter Context Acknowledge).

Il n'y a pas d'établissement de connexion de présentation proprement dite. Les protocoles CP, CPA, CPR sont codés dans les champs «user data» de la connexion de session *(fig. 10,* P-Connect, control information), tandis que les protocoles AC et ACA sont transportés comme paramètres des données «typed data» de la session. Même s'ils ne font pas partie de l'application, ces paramètres sont spécifiés en utilisant la syntaxe ASN.1 (X.208) et peuvent être codés par les règles de codage d'ASN.1 (X.209).

Il faut souhaiter que les mécanismes très complexes mis sur pied par les normateurs (DCS, etc.) se décantent au cours du temps et avec l'expérience acquise et qu'une tour de Babel des méta-langues et des syntaxes de transfert ne soit pas à nouveau construite.

3 L'étage de l'application

31 La structure logique des applications

Dans l'environnement OSI, la communication entre les processus d'application est représentée comme communication entre paires d'entités d'application (Application Entities AE). Ces entités (les officiers) sont décomposées en unités logiques séparées, auxquelles correspond généralement un processus logiciel ou, en terme de protocole, une machine protocolaire (protocol machine). Ils représentent, dans notre analogie, les processus mentaux des officiers. Ces unités logiques se décomposent en *(fig. 7* et *9):*

- un élément d'usager (user element) qui représente la partie du processus de l'application hiérarchiquement le plus élevé. Il fait usage des éléments logiques (ASE) pour accomplir l'application demandée. Il caractérise, en quelque sorte, le rôle de l'officier dans l'application en question (officier-automobile dans l'analogie)
- une collection d'éléments de service de l'application (Application Service Elements ASE) qui peuvent s'appeler les uns les autres pour accomplir leur fonction. Chacun de ces éléments a en principe sa propre syntaxe abstraite. C'est comme si chaque comportement préprogrammé de l'officier était signifié par une méta-langue conceptuelle propre.

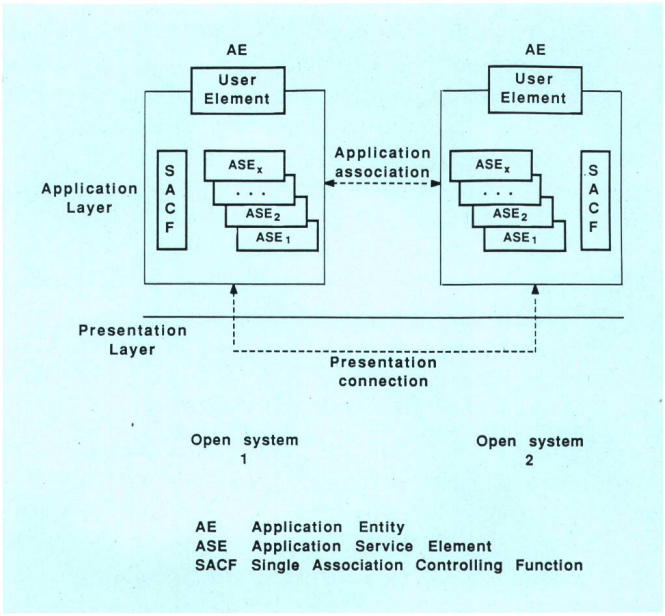

Fig. 7
Application structure de niveaux
AE Application Entity
ASE Application Service Element
SACF Single Association Controlling Function

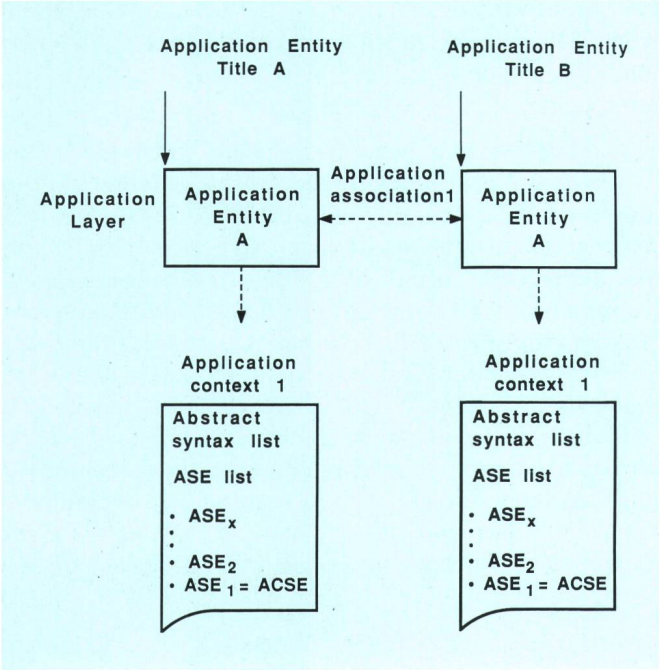

Fig. 8
Eléments d'application

Une unité particulière, appelée fonction de contrôle d'association simple (Single Association Controlling Function), devrait assurer le bon fonctionnement entre tous ces processus. Dans l'analogie, il doit y avoir un rapport certain entre le bon fonctionnement de cette unité et le quotient intellectuel de l'officier.

Il semble que les implantations de logiciels ne pourront utiliser ces concepts tels quels, et que certaines machines protocolaires pourraient être réalisées en commun, alors que d'autres ne seraient que des artifices de description, vides de sens pour l'implémenteur. La description de ces processus d'application laisse encore bien à désirer et subira certainement quelques modifications.

Trois concepts sont importants pour la couche d'application *(fig. 8)*:

- l'association d'application (Application-Association)
- le contexte d'application (Application-Context)
- le titre d'application (Application-Title).

L'*association d'application* est le lien entre deux entités d'application (AE), établi par la connexion de présentation. Tandis que cette dernière forme un pipe-line pour transférer les valeurs syntaxiques des données, l'association d'application spécifie les procédures agréées entre les entités d'application et les sémantiques partagées. Le processus mental inférieur de session a défini le genre de dialogue. Le processus mental suivant a choisi la langue véhiculaire (présentation). L'association d'application est le processus mental supérieur qui établit le lien entre deux rôles (officiers-automobiles).

Le *contexte d'application* est une liste d'éléments de service d'application (Application Service Elements) disponible aux entités d'application durant toute une association d'application pour leur interfonctionnement. Cette liste de machines protocolaires logiques définit le rôle d'une entité d'application dans un système ouvert (liste des comportements préprogrammés nécessaires pour tenir le rôle d'officier-automobile).

Le *titre d'application* est formé par les éléments d'information qui identifient de manière non ambiguë une entité d'application particulière sur un système ouvert bien déterminé quelque part dans le monde. C'est le nom précis, par exemple, de l'officier-automobile parlant l'allemand dans le poste de commandement No 2. Ces titres devraient devenir un des éléments des annuaires (directories), enregistré par une autorité de désignation, telle que les administrations.

32 Le service de contrôle d'association (CCITT X.217, X.227)

Le service de contrôle d'association (Application Control Service Element ACSE) est un élément de service d'association (Association Service Element) de base, c'est-à-dire qu'il devrait être utilisé systématiquement par toutes les spécifications de contexte d'application. Cet outil simple fournit les facilités de base pour le contrôle d'une association d'application entre deux entités d'application AE. Il définit le début et la fin de l'utilisation de l'association pour les procédures ASE identifiées dans le contexte d'application. C'est ce processus mental «supérieur» de base de l'officier qui établit le contact avec son partenaire (fig. 9) par l'intermédiaire du soldat de transmission.

Les machines protocolaires des éléments de service d'association échangent essentiellement les protocoles suivants d'application d'unités de données (APDU):

- Demande d'association A AARQ APDU (A-Associate-Request, fig. 10)

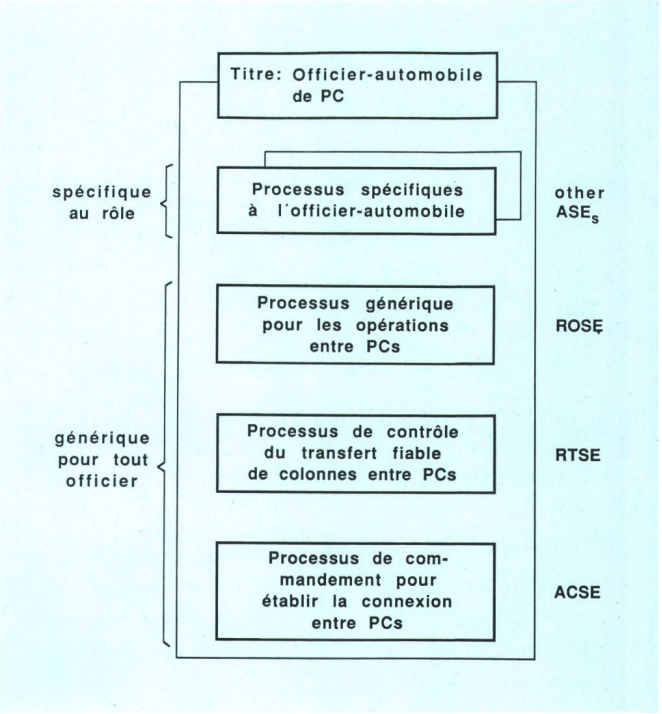

Fig. 9
Quelques processus mentaux de l'officier
ACSE Association Control Service Element
RTSE Reliable Transfer Service Element
ROSE Remote Operation Service Element
ASE Application Service Element

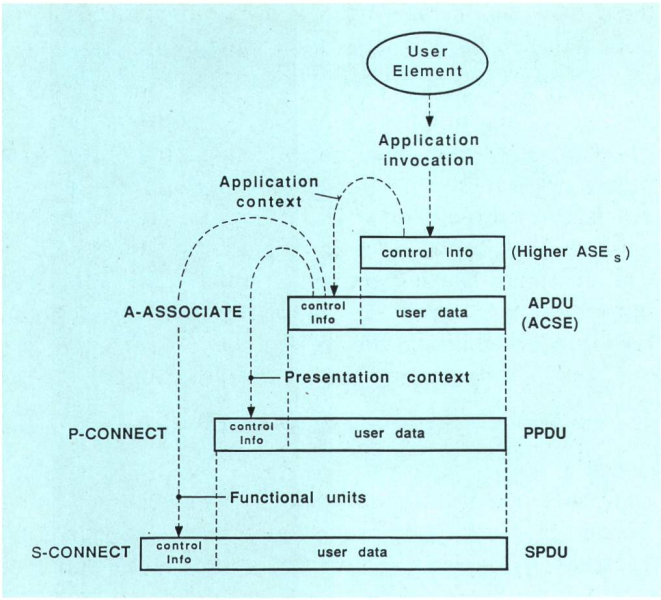

Fig. 10
Typical application association establishment
SPDU Session Protocol Data Unit
PPDU Presentation Protocol Data Unit
APDU Application Protocol Data Unit
ASE Application Service Element
ACSE Association Control Service Element (basic ASE)

- Réponse d'association A AARE APDU (A-Associate-Response)

Pour la libération de l'association, on a recours aux protocoles suivants:

- Demande de libération RLRQ APDU (A-Release-Request)
- Réponse de libération RLRE APDU (A-Release-Response).

Pour l'établissement de l'association, le protocole de demande d'association contient essentiellement (fig. 10, A-Associate control information):

- le nom du contexte d'application valable pour la durée de l'association (officier-automobile)
- le titre de l'application appelante (nom de l'officier du PC1)
- le titre de l'application appelée (nom de l'officier du PC2).

Pour la réponse, le protocole de réponse d'association contient, en plus du contexte d'application, le titre de l'application effectivement répondante (lorsque ce n'est pas l'officier-automobile Meier, mais l'officier-automobile Müller qui répond).

L'élément de service de contrôle d'application (ACSE) possède sa propre syntaxe abstraite. A l'état actuel, c'est par la seule syntaxe abstraite définie (ASN.1) que les protocoles d'application d'unités de données (APDU) de ce service de contrôle d'association sont décrits. Les règles de codage selon X.209 leur sont applicables.

L'officier pourrait ouvrir d'abord une connexion de session avec son partenaire, par laquelle il négocierait les unités fonctionnelles de session, puis ouvrir une connexion de présentation où il négocierait la langue véhiculaire (contexte de présentation), et enfin ouvrir une association d'application avec l'outil mental ACSE, où il négocierait le contexte d'application.

En fait, dans les applications actuelles, les trois couches supérieures sont ouvertes *en même temps* (fig. 10), après un travail mental de l'officier. L'officier-automobile décide (user element) de lancer une opération appelée ROSE entre le PC1 et le PC2. Ce processus mental supérieur passe l'argument de l'opération ROSE au processus inférieur ACSE, en lui indiquant à quel titre il veut établir l'association d'application (contexte d'application). Vu le partenaire qu'il désire atteindre dans le PC2, son mental «linguistique» choisit la langue véhiculaire qu'il a l'intention d'utiliser (contexte de présentation). Puis, en fonction du rôle et de l'opération à accomplir, il choisit les procédures de dialogue qu'il voudrait employer avec son partenaire (unités fonctionnelles de session).

L'officier donne l'ordre de connexion de session (envoi du premier char opérationnel de transport), avec tous les détails de session, de présentation et d'application contenus dans cette unité de protocole (Session Connect-Session Protocol Data Unit/SPDU, fig. 10), seulement lorsque son mental a mis de l'ordre dans toutes ces idées.

33 Le service de transfert assuré (CCITT X.218, X.228)

Le service de transfert assuré, appelé «Reliable Transfer Service Element (RTSE)», est une brique ou plutôt un processus mental supplémentaire dans la couche d'application. Cet outil forme un élément de service d'application (ASE) pour lui-même et possède sa propre syntaxe abstraite associée. Il n'est pas utilisé dans tous les cas par l'application.

Si la pile de sept couches d'une connexion OSI demande une classe de transport solide (classe 4), qui permet de numéroter les protocoles d'unités de données de transport et de faire de la reprise de bout en bout entre systèmes ouverts au niveau du transport, le service de transfert assuré est inutile (cas se produisant, par exemple, sur les réseaux locaux actuels, *fig. 11a)*. L'officier n'a pas à faire de reprise en cas d'erreurs, parce qu'il fait confiance au soldat de transmission.

Si, d'autre part, une application utilisant comme couche de transport associé la couche minimale (classe 0) accepte qu'en cas d'erreur lors du transfert des protocoles d'application des unités de données (APDU) d'un système ouvert à un autre il n'y a pas de reprise possible et qu'il faille recommencer le transfert dès le début en cas de problème, le service de transfert assuré est également superflu *(fig. 11b)*. Dans l'analogie de l'article antérieur, dès qu'un char tombe en panne, toute la colonne de chars doit être à nouveau envoyée, même si le char défectueux est le dernier. La session est aussi réduite à son strict minimum (noyau/kernel par exemple). En effet, les mécanismes de synchronisation et de reprise sont laissés de côté et on peut se demander, à juste titre, quel est le sens réel des services des couches cinq et quatre d'OSI, par rapport au prix à payer en lour-

deurs, tant du point de vue administratif que hiérarchique.

En revanche, le service de transfert assuré a toute sa raison d'être lors de transfert de protocoles d'application d'unités de données (ADPU) de gros volume (kilo-octets à mega-octets par APDU, par exemple). Le RTSE isole les autres ASE de l'application par rapport à la session et assure pour ceux-ci un mécanisme de reprise en cas d'erreur de transfert dans un APDU qui minimise la quantité d'informations à retransmettre. Après le passage transparent des APDU par le RTSE, les autres ASE ont la garantie du transfert complet et unique de chacun des APDU entre les deux systèmes ouverts. Un APDU de quelques mega-octets peut sembler énorme. Pourtant il suffit de penser au volume de transmission d'un document A4 en mode fac-similé groupe 4 pour comprendre que ces volumes sont très réalistes *(fig. 11c)*. Dans ce dernier cas, l'officier (application) s'assure non seulement contre les erreurs du réseau, mais aussi contre celles de son soldat de transmission (logiciel de couche 4). Il a ainsi la garantie et la quittance directe d'officier à officier, de mental à mental, sans passer par son subalterne. Ainsi en Télétex, l'officier envoie des points de synchronisation mineurs après chaque page de document et contrôle personnellement que le document total est bien reçu (point de synchronisation majeur), même si le soldat peut compter sur le service de reprise en cas d'erreurs au niveau paquet X.25 du réseau.

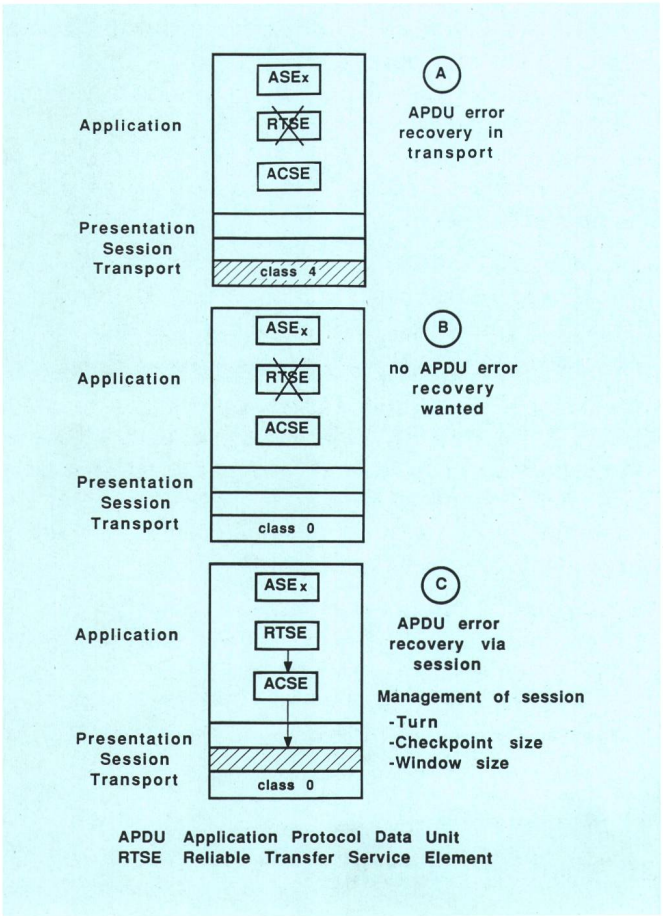

Fig. 11
Utilisation du service de transfert assuré
APDU Application Protocol Data Unit
RTSE Reliable Transfer Service Element

Le service de transfert assuré (processus mental supérieur) utilise les ressources de la session (processus mental inférieur) et contrôle le dialogue en définissant essentiellement:

- le mode de dialogue à utiliser (monologue, deux voies à l'alternat)
- le contrôle du «TURN», pour savoir qui a le droit de démarrer et de terminer l'association d'application et indirectement la session
- la grandeur du «checkpoint size», c'est-à-dire la quantitié maximale de données d'application (en unités de 1024 octets) qui peut être envoyée avant que la session introduise un point de synchronisation mineur (minor snychronization point) dans l'échange
- la fenêtre de contrôle d'erreur (window-size), qui définit le nombre maximal de points de synchronisation mineurs non quittancés que la session aura le droit d'envoyer avant d'attendre une quittance et de suspendre tout transfert durant cette attente.

Le service de transfert assuré utilise à son tour le service de contrôle d'association (ACSE) pour l'établissement et la libération de l'association d'application (fig. 11c).

34 Le service d'opération distante (CCITT X.219, X.229)

341 Le concept d'opération distante

Le service d'opération distante, ou «Remote Operation Service Element (ROSE)», est un outil de plus dans la panoplie de la couche application (mental de l'officier); c'est le plus sophistiqué. Cet outil sert essentiellement à la mise en œuvre d'applications interactives dans un environnement réparti. Une opération distante est demandée par une entité d'application (officier); son partenaire tente de l'exécuter et informe du résultat, si désiré (fig. 9).

L'outil ROSE n'est pas seulement une machine protocolaire comme RTSE et ACSE, il constitue une notation générique d'opérations, qui permet de définir des éléments de service propres à l'application désirée *(fig. 12)*.

La force de l'outil ROSE réside dans le fait qu'il est à même de définir une opération distante complète (avec résultats, si désiré) en utilisant une seule et unique «Macro-operation» définie en syntaxe ASN.1. L'invocation spécifique d'une telle macro-opération permet de faire d'une pierre trois coups; il est ainsi possible de

- décrire le service fourni par l'opération
- dériver la syntaxe des protocoles d'application d'unités de données (APDU) nécessaires à réaliser cette opération
- dériver également le séquence protocolaire de l'échange de ces APDU.

Il s'agit certainement d'un outil générique qui a un bel avenir. Dans notre analogie avec l'officier, ce n'est pas tant un processus mental préprogrammé, qu'une *marche à suivre générale pour toutes les «opérations»*, au sens militaire du terme, entre les deux PC. C'est le vade-mecum de tout officier de PC. Cette marche à suivre est applicable à tout type d'opération, qu'il s'agisse de transfert d'informations, de véhicules, de munitions,

etc. Il suffira lors de l'opération concrète (application désirée) d'introduire les paramètres en fonction des besoins propres.

Une opération distante peut être analysée en fonction du rapport attendu après l'exécution (pas de rapport, ou rapport en cas d'erreur, rapport en cas de succès, rapport dans tous les cas). Elle peut être divisée aussi en différents modes d'opération:

- le mode synchrone. L'entité qui invoque l'opération attend une réponse avant d'invoquer une autre opération
- le mode asynchrone. L'entité qui invoque peut demander d'autres opérations, avant d'avoir reçu les réponses des opérations précédentes.

Une autre facette des opérations distantes concerne la symétrie:

- en mode symétrique, les deux éléments de service de l'application peuvent invoquer la même collection d'opérations (officiers de même niveau hiérarchique)
- en mode asymétrique, l'un des éléments de service joue le rôle de fournisseur (supplier) et l'autre le rôle de consommateur (consumer, officiers de niveau hiérarchique différent).

On peut compliquer encore le modèle, en couplant certaines opérations entre elles, l'une étant génitrice (parent-operation), et les autres formant la descendance (child-operations).

342 La notation de commande à distance (Remote Operation)

La notation de commande à distance (Remote Operation) est basée donc sur le concept des macro-opérations définies en ASN.1 (X.208). Elle se fonde sur les principes de programmation classiques, ce qui devrait permettre de développer des outils automatiques pour intégrer les applications définies par cette notation dans l'environnement d'exécution de ces applications. Quatre opérations de ce genre sont définies:

- BIND. Macro-opération permettant de spécifier, pour l'élément d'usager (user-element) de l'application, l'opération qui établira l'association d'application. C'est dans cette opération que l'élément d'usager pourra définir des paramètres du contexte d'application à passer aux processus mentaux subordonnés. C'est la valeur de l'argument de l'opération BIND qui est en principe contenue dans le champ «user data» de la demande d'association A (fig. 10).
- UNBIND. Macro-opération permettant de spécifier l'opération de libération d'association qui sera invoquée par l'«user-element».
- OPERATION. Macro-opération permettant de spécifier les opérations qui formeront les éléments de service d'application invoqués par l'«user-element», une fois l'association d'application établie.
- ERROR. Macro-opération pour les cas d'exception.

Le résultat d'une opération est attendu sur la même association d'application que l'invocation. Il n'y a, en principe, pas de libération d'association avant que toutes les opérations invoquées soient terminées.

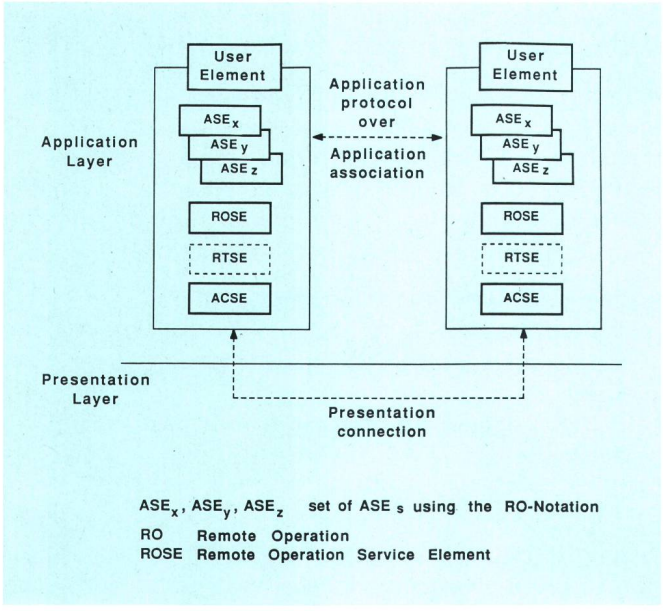

Fig. 12
Utilisation de l'élément de service de commande à distance
ASE_x, ASE_y, ASE_z valeurs des éléments de service d'application ASE_s utilisant la notation RO
RO Remote Operation
ROSE Remote Operation Service Element

343 Eléments de protocole

Les macros-OPERATIONS génèrent en principe quatre unités de données du protocole d'application (APDU):

- ROIV APDU invocation de l'opération par un officier
- RORS APDU transfert du résultat positif de l'opération par l'officier opposé
- ROER APDU transfert d'une information d'erreur, dans le cas d'un résultat négatif de l'opération
- RORJ APDU refus d'une requête (invocation) ou d'une réponse (résultat ou erreur) d'autres entités d'application.

Ces unités de données sont transférés en tant que données d'usager (user data), soit par le biais du service de transfert assuré (RTSE), s'il est utilisé, soit directement par l'entité de présentation.

Un exemple d'utilisation de ces processus mentaux (ACSE, RTSE et ROSE) dans le concept de la messagerie électronique est expliqué dans un prochain article. Ainsi, dans le protocole entre le «User Agent» et son «Message store» associé, trois éléments de service d'application supérieurs utilisent ROSE:

- l'un pour la gestion des paramètres entre les deux
- l'autre pour le dépôt des messages de l'«User Agent» par l'intermédiaire du «Message Store»
- et le dernier pour la recherche des messages reçus dans le «Message Store».

4 Conclusions

41 Synthèse (CCITT X.220)

Après ce bref aperçu des couches 6 et 7 et de leurs outils, il est bien nécessaire de faire une synthèse des couches supérieures OSI (4...7); la norme CCITT X.220 tente de le faire. C'est une description de l'utilisation

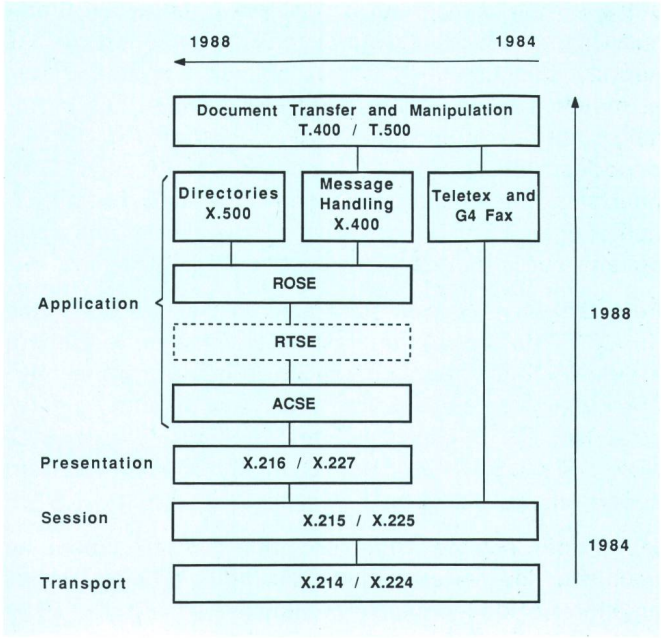

Fig. 13
Evolution des piles de protocoles

des «piles» de protocoles dans les applications du CCITT (OSI Protocol Stacks). Elle démontre que les services ne peuvent pas attendre la définition totale du gratte-ciel OSI pour être opérationnels et, donc, que plus les outils généraux des sept couches se consolident, plus ils sont utilisés pour définir les services finaux commercialisés *(fig. 13)*. Ainsi:

- le service Télétex, normalisé en 1980, par lequel il a fallu d'énormes efforts pour que les couches session et transport, mal stabilisées à l'époque, soient compatibles à celles de 1984 et de 1988 (CCITT T.62, T.70)
- le service «Message Handling», normalisé en 1984, où il a fallu définir des artifices de compatibilité vers le bas, pour assurer l'interfonctionnement entre les versions 1984 et 1988, vu qu'en 1984 la présentation et le concept des éléments de service de contrôle des applications n'étaient pas définis (CCITT X.400)
- le service «Directory», normalisé en 1988, dans lequel on utilise tous les outils définis actuellement dans l'application (CCITT X.500)
- l'architecture de document (CCITT T.400/T.500), utilisant ASN.1.

On peut donc assez facilement prévoir l'évolution et la complexité toujours plus grande du processus, à l'image des transformations de notre société de haute technologie.

42 Remarques critiques

La construction des piles OSI de protocoles est un processus dynamique qui n'est certainement pas arrivé à son terme. D'énormes progrès ont été faits, et ces «piles» sont déjà utilisées pour offrir des services commerciaux. Il est toujours plus facile de critiquer un gratte-ciel après sa construction plutôt qu'avant. Cependant certains points devraient encore se décanter:

- le formalisme à outrance de l'application, avec le jeu théorique trop poussé des multiples syntaxes abstraites
- le rôle réel qui sera donné aux couches 5 et 6 dans le futur et si, maintenant que nous commençons à être confrontés avec de vraies applications, le modèle ne va pas se décanter en trois macro-couches:
  - le dessous du transport
  - le transport (soldat)
  - l'application (officier)
- la façon de décrire et de faire travailler les multiples processus plus ou moins dépendants et plus ou moins autonomes de la couche d'application dans un environnement réparti
- la façon d'ouvrir les systèmes lorsqu'il y a plus de deux PC, c'est-à-dire deux «Open systems» en jeu.

43 Les enjeux

Pour reprendre l'analogie du verre à vin remarquons que ce verre a été rempli d'un vin de grand millésime ... l'arôme s'en dégage déjà, mais il faut encore le chambrer. Le précieux liquide sera prêt d'ici peu à être bu, c'est-à-dire utilisé dans notre environnement quotidien. Voyons un peu comment déguster ce précieux breuvage, c'est-à-dire comment les décisionnaires responsables de réseaux de communication doivent maintenant sérieusement évaluer l'applicabilité d'OSI dans leur propre contexte. Voici quelques éléments de décision *(fig. 14* et *15)*.

Limite de responsabilité

Dans le cas où deux systèmes informatiques doivent échanger des données en ligne (on line) et de manière très contrôlée, les piles de protocoles OSI permettent cet échange. Un échange qui ne peut se faire selon l'idée: «Ouvrir chacun des systèmes le plus possible», mais plutôt selon le principe d'«Ouvrir les systèmes juste ce qu'il est nécessaire». Ainsi OSI est à mettre en œuvre entre deux entreprises, ou encore au sein d'une même entreprise, entre deux centres de calcul dépendants de deux unités organisationnelles différentes, avec deux équipes différentes d'exploitation et d'opération ... ou deux cultures informatiques différentes (IBM, DEC, etc.).

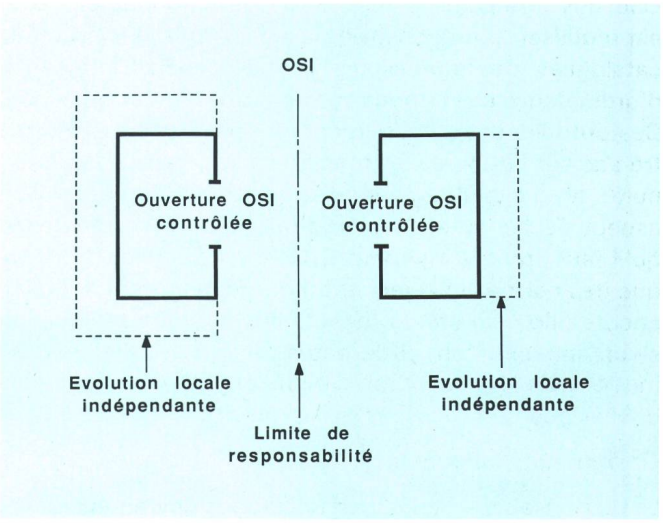

Fig. 14
Limite de responsabilité et OSI

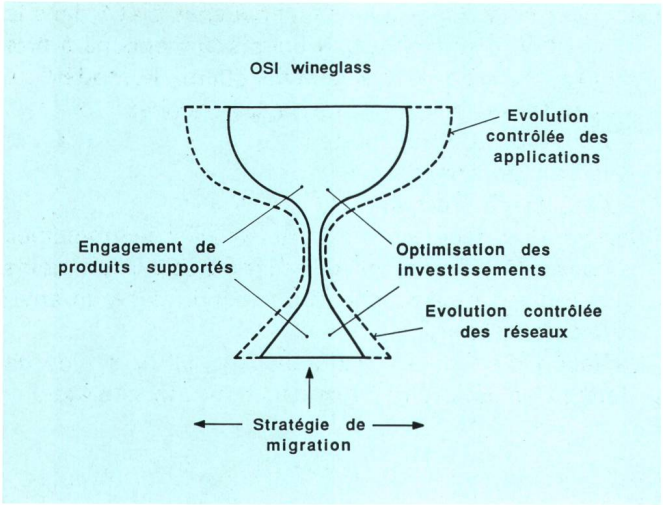

Fig. 15
Gestion du changement et OSI

Cette interface permet une évolution indépendante locale de chacun des systèmes (nouvelles machines, nouveaux systèmes opérationnels, etc.) sans que l'ouverture OSI sur le monde en soit perturbée (fig. 14).

Evolution contrôlée de la barrière de responsabilité

L'architecture OSI permet une migration contrôlée de la barrière de responsabilité, tout en assurant la continuité de l'interfonctionnement. Elle permet de supprimer ou de rajouter des jeux de protocoles, soit dans l'application, soit dans les services de transport, de telle sorte que le fait de rajouter une application (haut du verre OSI) ne détruise pas la base (pied du verre OSI), et inversement, si un nouveau type de réseau est rajouté (fig. 15).

Optimisation des investissements

Les étages OSI (mis à part la couche 1 et une partie de la couche 2) sont tous réalisés en logiciel. L'utilisation de l'architecture OSI permet donc d'optimiser les coûts de développement, et surtout d'exploitation et de maintenance des progiciels de communication, par une migration contrôlée des couches de logiciels (fig. 15).

Utilisation de produits existants

Une des stratégies d'application de l'architecture OSI est d'utiliser, pour construire les piles OSI, des produits catalogués de fournisseurs professionnels (fabricants d'ordinateurs ou grandes maisons de logiciels, fig. 15). Ces produits ont certainement un retard de deux à quatre ans sur l'état des normes, mais la garantie de continuité et d'évolution vers les normes futures est un aspect de sécurité dans le choix des investissements qu'il faut prendre en compte. Cela est d'autant plus vrai que des normes qui viennent d'être définies contiennent encore une quantité de défauts mineurs que seules plusieurs implantations différentes testées entre elles permettent de découvrir et d'éliminer. Et il ne faut pas oublier que ce processus prend plusieurs années.

Gestion du changement

Le responsable en télécommunications devrait avoir l'intuition que le moment est arrivé, où il doit exiger de ses fournisseurs des normes OSI pour réaliser l'interface entre le monde extérieur à son centre de calcul. Il doit aussi comprendre le dynamisme et l'évolution de ces normes, ainsi que des produits logiciels correspondants, et mettre sur pied, avec son fournisseur et ses partenaires, un plan d'*évolution* et de *migration* (fig. 15). Les périodes d'étude du CCITT sont de quatre ans. Elles pourraient être une dimension de la fréquence de migration et d'évolution nécessaires s'il veut garder une porte ouverte sur le monde.

Pour terminer, l'auteur tient à remercier ses collègues du CCITT qui l'ont aidé à pénétrer dans cette matière ardue et lui ont permis de saisir un peu la «substantifique moelle» de ces normes. Il convient de citer en particulier MM. *J. B. Steffani* (France), *Peter Ingram* (UK), *Jim White* (USA), *Douglas Steedmann* (Canada) et, tout spécialement, M. *Pietro Schicker* (CH).

Merci enfin à Mme *Andersen,* qui a pris la patience de déchiffrer les textes initiaux des deux articles, de les améliorer et de les mettre au net.

Bibliographie

- CCITT/88 X.200. Reference model of open systems interconnection for CCITT applications.
- CCITT/88 X.208. Specification of abstract syntax notation one (ASN.1).
- CCITT/88 X.209. Specification of basic encoding rules for abstract syntax notation one (ASN.1).
- CCITT/88 X.210. Open system interconnection (OSI) layer service conventions.
- CCITT/88 X.211. Physical service definition for open systems interconnection for CCITT applications.
- CCITT/88 X.212. Data link service definition for open systems interconnection for CCITT applications.
- CCITT/88 X.213. Network service definition for open systems interconnection for CCITT applications.
- CCITT/88 X.214. Transport service definition for open systems interconnection for CCITT applications.
- CCITT/88 X.215. Session service definition for open systems interconnection for CCITT applications.
- CCITT/88 X.216. Presentation service definition for open systems interconnection (OSI) for CCITT applications.
- CCITT/88 X.217. Association control service definition for open systems interconnection (OSI) for CCITT applications.
- CCITT/88 X.218. Reliable transfer: Model and service definition.
- CCITT/88 X.219. Remote operations: Model, notation and service definition.
- CCITT/88 X.220. Use of X.200-series protocols in CCITT applications.
- CCITT/88 X.223. Use of X.25 to provide the OSI connection-mode network service for CCITT applications.
- CCITT/88 X.224. Transport protocol specification for open systems interconnection for CCITT applications.
- CCITT/88 X.225. Session protocol specification for open systems interconnection for CCITT applications.
- CCITT/88 X.226. Presentation protocol specification for open systems interconnection (OSI) for CCITT applications.
- CCITT/88 X.227. Association control protocol specification for open systems interconnection (OSI) for CCITT applications.
- CCITT/88 X.228. Reliable transfer: Protocol specification.
- CCITT/88 X.229. Remote operations: Protocol specification.
- CCITT/88 X.244. Procedure for the exchange of protocol identification during virtual call establishment on packet switched public data networks.
- CCITT/88 X.290. OSI conformance testing methodology and framework for CCITT applications.
- *Pitteloud J.* Le fond du verre à vin OSI, Techn. Mitteilungen PTT, No. 1/1989
- *Pitteloud J.* Der Fuss des OSI-Weinglases, Techn. Mitteilungen PTT, No. 2/1989