

Chapitre 9 : Le niveau Présentation

/udd/bcousin/Pages-web-Armor/Enseignement/Reseaux-generalites/Cours/9.fm - 1 juin 2001 18:24

Plan

- 1. Introduction p311
- 2. Notation de syntaxe abstraite p320
- 3. Les règles d'encodage p330
- 4. Protocole p339
- 5. Conclusion p343

Bibliographie

- D. Dromard & al., Réseaux informatiques : cours et exercices, Eyrolles, 1994. Tome 2, chapitre 4.
- H. Nussbaumer, Téléinformatique, Presses polytechniques romandes, 1991. Tome 3, chapitre 2.
- A. Tanenbaum, Réseaux, InterEditions, 1997. Chapitre 7.

1. Introduction

1.1. Présentation

La 6^{ème} couche !

Garantir le contenu informationnel (**sémantique**) de l'ensemble des données échangées.

Nécessite un moyen de description des structures de données :

- souple (adapté à toutes les applications)
- normalisé (compatible, interopérabilité)

1.2. Les problèmes

1.2.1 Introduction

Le transfert **transparent** (c'est-à-dire à l'identique) de la suite binaire n'est pas le service attendu !

En effet une même suite binaire peut être interprétée de manières multiples :

- ex : 0x31
 - . soit code ASCII de 'a'
 - . soit représentation en complément à 2 du nombre décimal 49

☞ Il faut un transfert de données qui conserve la **sémantique**.

On va transmettre non seulement la **valeur** de la données mais aussi

- son **type** ☞ la technique de représentation utilisée !

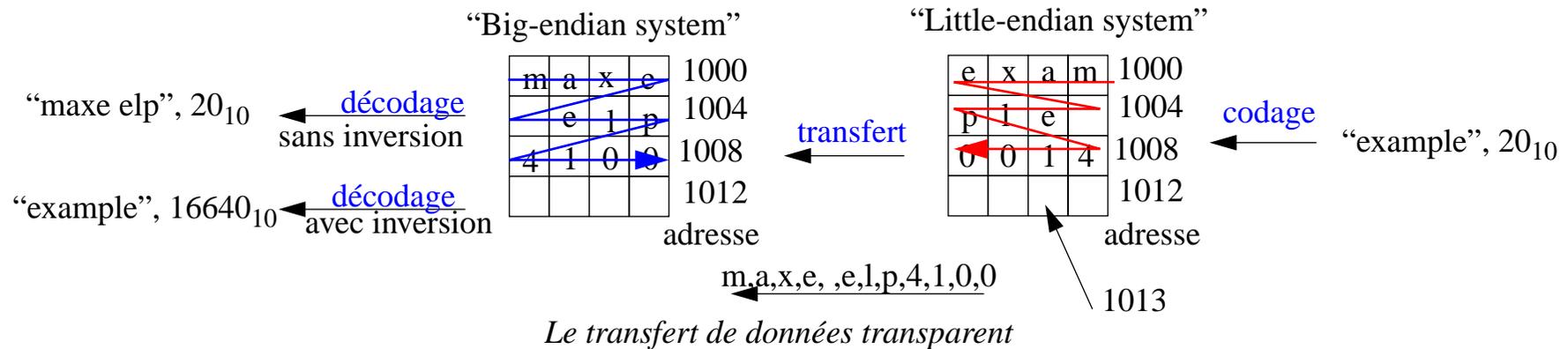
Plusieurs facteurs sont à prendre en compte :

- l'hétérogénéité des machines
- les différentes techniques de représentation
- les objets complexes
- la longueur et le nombre instances de l'objet

1.2.2 L'hétérogénéité des architectures internes des ordinateurs

Si deux ordinateurs ayant des architectures différentes échangent des données à l'aide d'un service de transfert qui conserve totalement la suite binaire, l'interprétation des données reçues risque d'être incorrecte.

- Méthode de placement des mots en mémoire :
 - . Intel, DEC/Motorola, IBM
 - . "little endian/ big endian war" (voyages de Gulliver [Swift])



- La longueur des mots (2, 4, 8 octets), la longueur des caractères (7, 8, 16, 32 bits), etc.

1.2.3 La représentation des types simples

- technique de représentation des caractères :
 - . EBCDIC, ASCII, etc.
- technique de représentation des nombres entiers :
 - . représentation binaire en complément à 1, ou complément à 2, DCB, etc.
- technique de représentation des nombres réels :
 - . virgule flottante, mantisse + exposant, base décimale ou binaire, etc.

1.2.4 Les objets complexes (composés)

Les applications échangent des objets simples mais aussi des objets structurés :

- exemple : liste, ensemble, tableau, vecteur, matrice, enregistrement, etc. Il faut être capable de transmettre la structure !

1.2.5 Echanges en nombre et en ordre variable

Le nombre, l'ordre et le type des objets échangés dépend de l'état de l'émetteur sans que le récepteur soit préalablement mis au courant !

- il faut être explicite : codage TLV

1.2.6 Spécification

Les applications réparties peuvent être développées non seulement sur des équipements hétérogènes mais aussi par des équipes différentes avec des moyens variés :

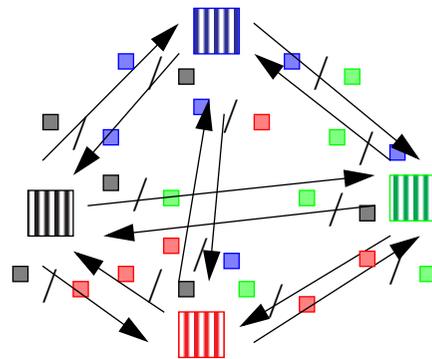
- les structures de données doivent être **spécifiées** préalablement et formellement,
- **indépendamment** de la représentation choisie (l'implémentation),
- **compréhensibles** par le développeur de code (humain)

1.3. Les solutions

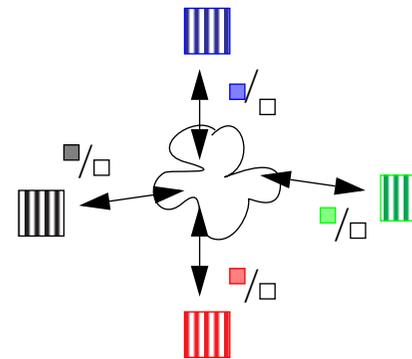
1.3.1 Présentation

Des fonctions de décodage tenant compte de la représentation utilisée :

- dépendent des couples émetteur/récepteur
- chaque station doit posséder un jeu complet de fonctions de décodage, si elles veulent communiquer avec n'importe quelles autres stations :
- N types de représentation $\rightarrow N*N-1$ fonctions de décodage



Fonctions de décodage



Une représentation commune

Une seule et unique représentation conventionnelle :

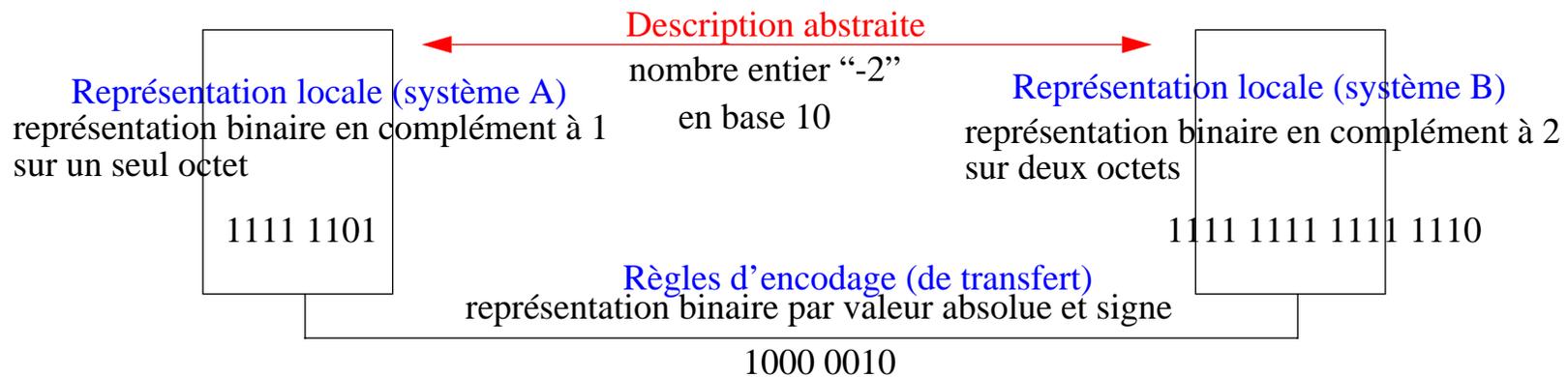
- définie par le réseau
 - . solution proposée par Internet et XDR

- tous doivent s'y plier, même si c'est sous-optimal !
 - . par exemple : si l'émetteur et le récepteur utilisent la même représentation interne qui n'est pas la représentation conventionnelle, il y a deux opérations de transcodage inutiles lors de chaque échange.
 - . trop rigide : pas d'adaptation vis-à-vis des applications vis-à-vis des évolutions futures

Spécification/Implémentation

On distingue trois types de syntaxe :

- la syntaxe abstraite, commune, adaptée à l'être humain, partagée par les entités
- la syntaxe concrète, dépendante et locale à chaque système (hors OSI)
- la syntaxe de transfert (Règles d'encodage), pour la transmission sur le réseau



1.3.2 La normalisation

La norme définit :

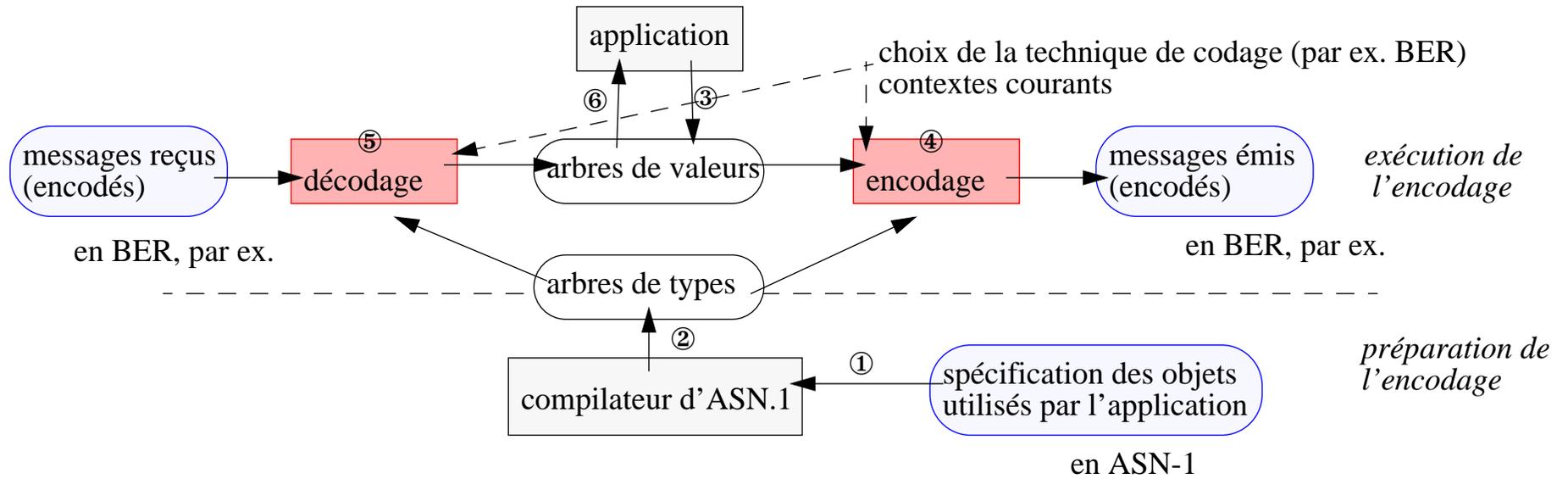
- un langage abstrait de spécification de la structure des données échangées
 - . ex : ASN.1
 - . interprétable par l'être humain (-> texte d'un programme)
 - . notion de type
 - . ressemble aux définitions de structures de données des langages évolués
 - . exemple :

numero INTEGER

- des techniques d'encodage (règles)
 - . plusieurs règles d'encodage : BER, PER, etc.
 - . c'est la suite binaire réellement transmise (-> code machine du programme)
 - . la technique d'encodage est choisie lors de l'établissement de la connexion
 - . exemple :

{numero 17} -> 02 01 11₁₆

L'utilisation :



2. Notation de syntaxe abstraite

2.1. Introduction

La notation de syntaxe abstraite [ASN.1](#) est un langage formel :

- définition des objets échangés entre entités homologues de niveau Présentation
- spécification de leur type ou structure
- exemple d'objets :
 - . les données de la couche supérieure (Application),
 - . les informations échangées pour l'administration du réseau Internet (SNMP)
 - . les N-PDU !
- de forme textuelle

Normalisé par le CCITT X.208 et ISO IS 8824.

La notation ASN.1 est basée sur la notion de [type](#) :

- similaire à celle des langages de programmation structurée

La notation de syntaxe abstraite ASN.1 est décrite sous la forme BNF (“Backus-Naur form”)

2.2. La notation

2.2.1 Présentation

Pour chaque objet échangé, on distingue :

- son **type**
- sa **valeur** (parmi l'ensemble des valeurs possibles du type)

Il existe des types simples et des types composés.

La norme ASN.1 définit des types prédéfinis :

- des types prédéfinis simples : INTEGER
- des types prédéfinis structurés : OCTET STRING

De nouveaux types (simples ou composés) peuvent être construits à partir d'autres types :

- grâce aux types : SET ou SEQUENCE (des objets constructeurs)

À chaque type est associé une étiquette (**Tag**) qui l'identifie :

- numéro de l'étiquette
- la classe de l'étiquette

On distingue quatre classes d'étiquettes (Tag class) :

- universelle
 - . types de base défini dans ASN.1
 - . ex : INTEGER, OCTET STRING, SET
- spécifique à une application
 - . associée à une application (ou à une classe d'application) particulière
 - . définie dans d'autres normes
- spécifique à un contexte
 - . la définition des types de cette classe ont une portée limitée au sein d'une partie d'une application
 - . permet de distinguer les éléments d'un ensemble
- privée
 - . définie par l'utilisateur, pour ses besoins propres.

Deux rôles d'ASN-1 :

- définition d'un type d'objet (rôle principal)
- définition de la valeur d'un objet (rôle secondaire)

2.3. Un exemple

2.3.1 La description informelle d'objets

L'enregistrement d'un employé :

Nom : Paul Durand
Emploi : professeur
Numéro d'employé : 51
Date d'embauche : 14 Juillet 1789
Nom de l'épouse : Anne-marie Martin
Nombre d'enfants : 2

de son premier enfant :

Nom : Marc Durand
Date de naissance : 11 Novembre 1914

de son deuxième enfant :

Nom : Paulette Dupont
Date de naissance : 8 Mai 1945

2.3.2 La description en ASN-1 de ces objets (leur valeur)

L'enregistrement :

```

{
  nom          { prenom "Paul", nom "Durand" },
  emploi      "professeur",
  numéroDemploye 51,
  dateDembauche 17890714090000.0,
  nomDeLepouse { prenom "Anne-marie", nom "Martin"},
  enfants     {
    {
      nom          { prenom "Marc", nom "Durand" },
      dateDeNaissance 1811110800Z,
    }
    {
      nom          {prenom "Paulette", nom "Dupont"},
      dateDeNaissance 450508070001Z,}
    }
  }
}

```

Automatic Tagging on !

2.3.3 La description en ASN.1 du type de ces objets

```

Employe DEFINITIONS ::= BEGIN
  EnregistrementEmploye ::= [APPLICATION 0] SET
  {
    nom          Nom,
    emploi       VisibleString,
    numeroEmploye NumeroEmploye,
    dateDemauche GeneralizedTime,
    nomDeLepouse Nom,
    enfants       SEQUENCE OF Enfant DEFAULT {}
  }

  Enfant ::= SET
  {
    nom          Nom,
    dateDenaissance UTCTime
  }

  Nom ::= [APPLICATION 1] SEQUENCE
  {
    prenom       VisibleString,
    nom          VisibleString
  }

  NumeroEmploye ::= [APPLICATION 2] INTEGER

END -- du module Employe

```

2.4. La syntaxe

Les commentaires :

- préfixé par --

Le module

- son nom, sa définition (DEFINITION)
- BEGIN/END
- IMPORTS/EXPORTS

La description de type :

- son nom, sa définition (::=)

La description de champ des types composés :

- leur nom, leur définition
- le nom sert lors de l'instanciation d'un objet de ce type a lui affecter une valeur

Type/nom : syntaxiquement les types commencent par une majuscule, les noms non !

Les type prédéfinis :

- INTEGER, VisibleString, UTCTime, GeneralizedTime, etc

Les types construits :

- simple : ex. NumeroDemploye
- composé : ex: Enfant

La classe des étiquettes (entre crochets) :

- applicative : [APPLICATION *n*]
- contextuelle : [*n*]
- universelle ou non-étiquetée : INTEGER ou *rien*
- privée : [PRIVATE *n*]

Les constructeurs :

- SET (OF), SEQUENCE (OF), CHOICE (!)
 - . ensemble/ensemble ordonné
 - . ensemble ayant des éléments de même type ou de types différents
 - . ensemble ayant un des types alternatifs proposés

Autres

- DEFAULT : valeur par défaut
- OPTIONAL : champ optionnel

- EXTERNAL : sélection de contexte
- OBJECT IDENTIFIER, ObjectDescriptor
- MACRO, TYPE NOTATION, VALUE NOTATION : macro-génération
- IMPLICIT : optimisation de l'encodage

2.5 - Les types universels de ASN.1

Tableau 1 :

UNIVERSAL Type	Code	Commentaire (constructeur)
BOOLEAN	1	
INTEGER	2	
BIT STRING	3	(O/N)
OCTET STRING	4	
NULL	5	donnée sans valeur
OBJECT IDENTIFIER	6	
ObjectDescriptor	7	
EXTERNAL	8	signale un changement de contexte (O)
REAL	9	
ENUMERATED	10	
SEQUENCE (OF)	16	2 versions (O)

Tableau 1 :

UNIVERSAL Type	Code	Commentaire (constructeur)
SET (OF)	17	2 versions (O)
NumericString	18	
PrintableString	19	
TeletexString	20	
VideotexString	21	
IA5String	22	codage ASCII
UTCTime	23	
GeneralizedTime	24	
GraphicString	25	
VisibleString	26	
GeneralString	27	

3. Les règles d'encodage

3.1. Introduction

Il existent plusieurs syntaxes de transfert :

- [Basic encoding rules](#) (ISO 8825/1, X.209)
- Canonical encoding rules (ISO 8825/1)
- Distinguished encoding rules (ISO 8825/1)
- Packed encoding rules (ISO 8825/2)

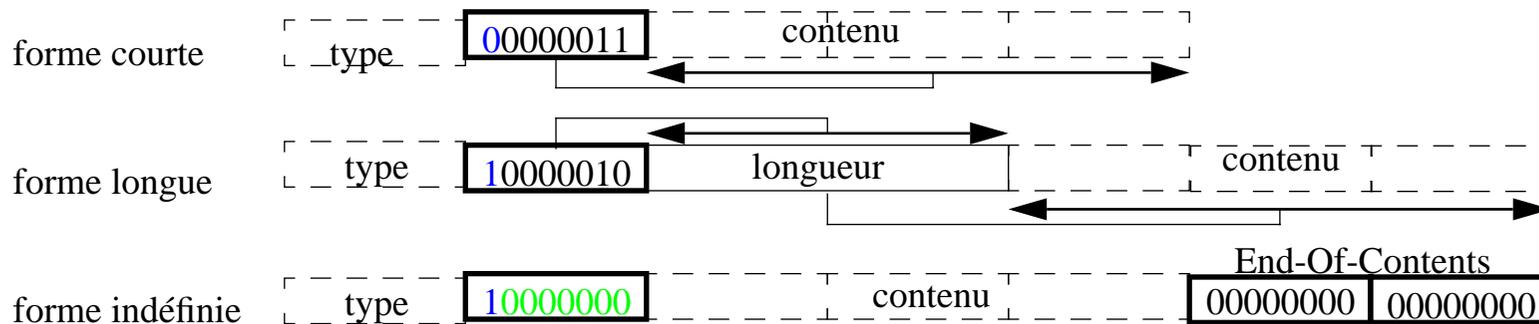
Codage TLV.

- explicite et flexible
- lourd

Nous allons présenter l'encodage le plus fréquent mais aussi le plus encombrant : [BER](#).

3.3. Le champ Longueur

- forme **courte**
 - . champ d'un seul octet : premier bit à 0
 - . longueur en octet du champ Valeur $< 2^7$
- forme **longue**
 - . champ sur plusieurs octets : premier bit du premier octet à 1
 - . premier octet : la longueur du champ longueur
 - . autres octets : longueur en octet du champ Valeur $\geq 2^7$
- forme **indéfinie**
 - . champ sur plusieurs octets : premier bit du premier octet à 1, les autres à 0
 - . longueur du champ valeur inconnue (lors du début de l'encodage)
 - . réservée aux objets composés (mais la longueur de chaque sous-objet est connue)
 - . terminée par un double octet nul (EOC : [UNIVERSAL 0], longueur = 0)



3.4. Encodage de la valeur des quelques types

3.4.1 Les types simples

Booléens : 1 octet

- FALSE : 00000000
- TRUE : $\langle \rangle 0$, par exemple 00000001
- Par exemple, encodage de TRUE :

Boolean	Length	Contents
01 ₁₆	01 ₁₆	FF ₁₆

Entiers :

- en complément à 2
- Par exemple, encodage de 51₁₀ :

Integer	Length	Contents
02 ₁₆	01 ₁₆	33 ₁₆

Les chaînes de bits :

- Par exemple, encodage de 040A3B5F291CD0₁₆ :

BitString	Length	Contents
03 ₁₆	07 ₁₆	040A3B5F291CD0 ₁₆

La chaîne d'octets ou de caractères

- suite d'octets, codage ASCII
- Par exemple, encodage de "Exemple !"

VisibleString	Length	Contents
1A ₁₆	09 ₁₆	"Exemple !"

3.4.2 Les réels

Trois formats d'encodage : base binaire, base décimale, valeurs particulières

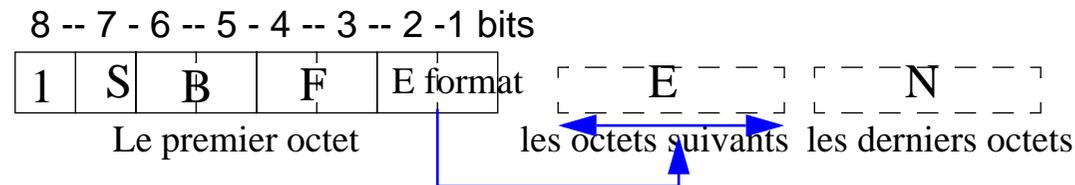
Le premier octet : la forme du codage :

- bit 8 = 1 : encodage en base binaire
- bit 8-7 = 00 : encodage en base décimale
- bit 8-7 = 01 : encodage de valeurs spéciales

L'encodage en base binaire de la valeur réelle R :

- Mantisse (M), Signe (S), Base (B), Facteur d'échelle (F), Exposant (E), valeur (N)
- $R = M \cdot B^E$, avec $M = S \cdot N \cdot 2^F$, $B \in \{2, 8, 16, \text{valeur réservée}\}$ et $F \in \{0-3\}$.

- Le premier octet du champ Valeur :



Encodage du champ valeur d'un réel en format binaire

- Les octets suivants : la valeur de l'exposant E par représentation binaire en complément à 2
 - . Eformat = 00 : le deuxième octet contient la valeur de l'exposant
 - . Eformat = 01 : les 2^{ème} et 3^{ème} octets contiennent la valeur de l'exposant
 - . Eformat = 10 : les 2^{ème}, 3^{ème} et 4^{ème} octets contiennent la valeur de l'exposant
 - . Eformat = 11 : le 2^{ème} octet donne le nombre d'octets suivants qui contiennent la valeur de l'exposant
- Les derniers octets :
 - . la valeur de la mantisse N en tant que nombre binaire non-signé
 - . la longueur du champ N est déterminée par la connaissance de la longueur totale du champ Valeur

L'encodage en base décimale :

- norme ISO 6093

- 3 formes possibles

L'encodage des valeurs spéciales :

- PLUS-INFINITY : premier octet = 01000000
- MINUS-INFINITY : premier octet = 01000001

Exemple :

- 09 04 C0 10 00 80 $\implies -2^7 \cdot 2^{16}$

3.5. Exemple

La définition ASN-1 de l'enregistrement a encodé :

```

Employe DEFINITIONS ::= BEGIN
    EnregistrementEmploye ::= [APPLICATION 0] IMPLICIT SET
    {
        nom                Nom,
        emploi              [0] VisibleString,
        numeroEmploye      NumeroEmploye,
        dateDembauche      [1] GeneralizedTime,
        nomDeLepouse       [2] Nom,
        enfants             [3] IMPLICIT SEQUENCE OF Enfant DEFAULT {}
    }

    Enfant ::= SET
    {
        nom                Nom,
        dateDenaissance    [0] UTCTime
    }

    Nom ::= [APPLICATION 1] IMPLICIT SEQUENCE
    {
        prenom             VisibleString,
        nom                VisibleString
    }

    NumeroEmploye ::= [APPLICATION 2] IMPLICIT INTEGER

END -- du module Employe
L'enregistrement encodé en BER :

```

Enreg.LengthContents : [Application 0]

60 8185

nom LengthContents

61 0E

*Visible String*LengthContents

1A 04 "Paul"

*Visible String*LengthContents

1A 06 "Durand"

emploi LengthContents

A0 10

*Visible String*LengthContents

1A 0A "Professeur"

numeroDemploye LengthContents

42 01 33

dateDembauche LengthContents

A1 0E

*GeneralizedTime*LengthContents

18 0E 17890714090000

nomDeLepouse LengthContents

A2 16

nom LengthContents

61 14

*Visible String*LengthContents

1A 0A "Anne-Marie"

*Visible String*LengthContents

1A 06 "Martin"

etc.

4. Protocole

4.1. Présentation

La couche Présentation est normalisée :

- Service : ISO IS 8822
- Protocole : ISO IS 8823

Offre les services de **transfert de données** offerts par la couche ... Session !

Et des **règles de représentation** des données ... **structurées**.

Et des mécanismes de **définition et de sélection** de ces règles.

4.2. Le service

Etablissement de la connexion :

- le contexte de présentation initial
- les caractéristiques de la connexion Session

Gestion des contextes

- définition des contextes de Présentation
- sélection du contexte courant

Transfert de données

- duplex, exprès, typés, de capacités (Session)

Libération de la connexion

- ordonnée, brutale (Session)

Contrôle du dialogue

- ceux fournis par la couche Session : point de synchronisation, activité, jeton

4.3. Le contexte

Un contexte de Présentation est défini par un couple :

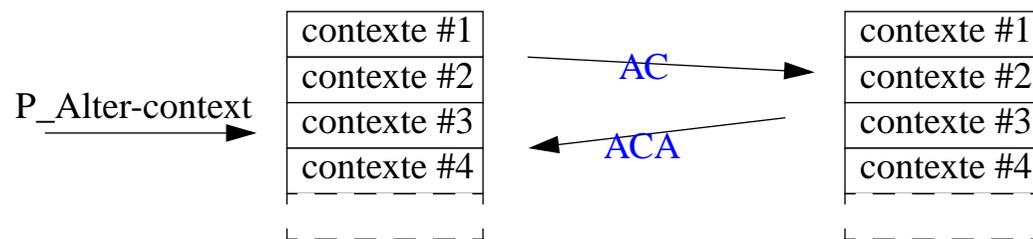
- une syntaxe d'application et une syntaxe de transfert

Plusieurs contextes sont disponibles :

- contexte par défaut
- contexte initial (négocié lors de la connexion)
- contexte sélectionné parmi un ensemble de contextes prédéfinis

Modification de l'ensemble des contextes prédéfinis :

- La primitive P__Alter-context



AC : Alter-context P-PDU, ACA : Alter-context Ack. P-PDU

4.3.1 Les P-PDU de gestion des contextes

Les P-PDU de contexte sont acheminés par le service de transfert des données typés de la couche Session

Les P_PDU AC contiennent :

- la liste des nouveaux contextes (identificateur de contexte, nom de syntaxe d'application, liste de syntaxe de transfert)
- la liste des contextes à supprimer (identificateur de contexte)

4.3.2 Sélection des contextes

Le type EXTERNAL permet de sélectionner un contexte courant.

Soit par référence indirecte (implicitement):

- identificateur de contexte

Soit par référence directe (explicitement)

- à l'aide d'un OBJECT IDENTIFIER
- par ex. : {joint-iso-ccitt asn1 (1) basic-encoding (1)} = ASN.1, BER

5. Conclusion

La couche Présentation

- s'appuie pour l'essentiel sur les services de la couche Session.
- Fonction de définition et de sélection de contexte

Contexte de la couche Présentation

- une syntaxe d'application (utilise la notation de syntaxe abstraite : ASN.1)
- une syntaxe de transfert (ex. BER)

Fonction de sécurité (cryptage, authentification, certification) et de compression des données

Un autre exemple : Internet

- basé sur un format conventionnel (universel),
- des fonctions d'encodage (ex : ntohs(), ntohl()),
- une norme XDR (eXternal Data Representation),
- un outil ("rpcgen")
- un protocole d'administration du réseau SNMP qui utilise ASN-1 et BER pour représenter et encoder les objets échangés !