

2.1 A notação usada em ASN.1

ASN.1 utiliza alguns tipos primitivos (elementos de dados) com os quais podem ser compostas estruturas mais complexas. Os tipos primitivos ou básicos são:

- BOOLEAN
- INTEGER
- BITSTRING
- OCTETSTRING
- NULL

Estruturas complexas são definidas agregando-se tais tipos primitivos de algumas formas previstas na ASN.1. As principais formas de estruturação de tipos compostos ou "construídos" são:

- SEQUENCE-lista ordenada de tipos
- SEQUENCE OF-iteração ilimitada de um único tipo
- SET-lista não ordenada de tipos
- SET OF-iteração ilimitada de um único tipo (a ordem não é importante)
- CHOICE-um campo que consiste de um valor dentre os tipos listados.

ASN.1 define uma notação para especificação de valores e para definição de tipos. A definição de um tipo consiste numa coleção de campos que, no mais baixo nível, consiste de um identificador, uma possível etiqueta (rótulo ou "tag"), uma referência e uma possível indicação de que aquele campo é opcional (pode ser omitido).

No menor nível os campos da definição são combinados usando mecanismos de estruturação que começa com o nome do tipo da estrutura e então, em geral, uma lista de campos separados por vírgulas e envolvidos em chaves "{}". É válido usar aninhamento de estruturas.

O identificador compreensível (letra minúscula) serve apenas para auxiliar a compreensão mas é ignorado pelas regras de codificação básicas.

O rótulo ("tag") permite que dois tipos aparentemente idênticos sejam separados tal que seu uso possa ser distinguido em construções tal como CHOICE ou SET.

Assim, um tipo será definido, segundo a ASN.1 usando-se uma das sequências de tipo válidas, que são:

```
Type ::= BuiltinType | DefinedType
```

```
BuiltinType ::= BooleanType |  
                IntegerType |  
                BitStringType |  
                OctetStringType |  
                NullType |  
                SequenceType |  
                SequenceOfType |  
                SetType |  
                SetOfType |  
                ChoiceType |  
                TaggedType |  
                AnyType |  
                CharacterSetType |  
                UsefulType |
```

O DefinedType especifica sequências externas usadas para referir definições de tipos e valores.

Os mais significativos tipos de primitivos serão definidos a seguir:

- O tipo **BOOLEAN** é usado para representar valores de variáveis lógicas (isto é que

somente podem assumir dois estados). Quando escolher um nome representar uma variável booleana deve ser escolhido um que descreva o estado verdadeiro, por exemplo:

Casado ::= **BOOLEAN**

- O tipo **INTEGER** é usado para modelar os valores designativos de ordem (cardinal) ou inteiros. Exemplo:

SaldoDaConta ::= **INTEGER** -- em centavos; negativo significa debito

Pode ser definido o valor mínimo e máximo, por exemplo:

DiaDoMes ::= **INTEGER** {primeiro(1),ultimo(31)}

Também pode-se usar inteiros como símbolos:

DiaDaSemana ::= **INTEGER** {Domingo(1), Segunda(2), Terça(3), Quarta(4), Quinta(5), Sexta(6), Sábado(7)}

EstadoMarital ::= **INTEGER** {solteiro(0),casado(1),viúvo(2)}

- Usa-se o tipo **BITSTRING** para modelar dados binários de formato e comprimento qualquer (inclusive não múltiplo do 8). Pode-se usar este tipo para representar um mapa em que se indica pela posição dos bits ligados a ocorrência de algum fato, por exemplo:

DiasComSolNoMes ::= **BITSTRING**{primeiro(1),ultimo(31)} -- Um dia foi ensolarado se o bit correspondente a ele no mapa tem valor 1

- O tipo **OCTET STRING** é usado para modelar dados de qualquer formato e comprimento múltiplo de 8 bits. Exemplo:

CorpoDoPacote ::= **OCTET STRING**

OU

Sobrenome ::= PrintableString

A segunda forma é preferível, quando possível.

- O tipo **NULL** é usado para indicar ausência de algum elemento numa sequência, tal como no exemplo seguinte:

```
IdentificaçãoPaciente ::= SEQUENCE
    { nome          OCTET STRING,
      numeroQuarto CHOICE
        {INTEGER,
         NULL --se o paciente já saiu do hospital-- }}

```

- O tipo **SEQUENCE OF** é usado para modelar uma coleção de variáveis cujo tipo é o mesmo, cujo número é grande e imprevisível e cuja ordem é significativa. Exemplo:

NomeNaçõesMembros ::= **SEQUENCE OF** ISO646String
-- na ordem em que se integraram

- O tipo **SEQUENCE** é usado para modelar uma coleção de variáveis cujo tipo é o mesmo, cujo número é conhecido e modesto e cuja ordem é significativa, desde que a composição do conjunto dificilmente varie de uma versão do protocolo para outra. Exemplo:

```
NomeDosFuncionarios ::= SEQUENCE
    {presidente      ISO646String,
     vicePresidente  ISO646String,
     secretaria      ISO646String }

```

Também pode ser usado o tipo **SEQUENCE** quando o tipo dos integrantes difere, mas o número é conhecido e modesto e cuja ordem é significativa, desde que a composição do conjunto dificilmente mude de uma versão do protocolo para outra. Exemplo:

```
Credenciais ::= SEQUENCE
{
  nomeUsuario  ISO646String,
  password     ISO646String,
  numeroConta  INTEGER }

```

Se os elementos de uma sequência são em número fixo mas de vários tipos, um nome de referência deve ser atribuído a cada elemento cujo objetivo não seja completamente evidente de seu tipo. Exemplo:

```
Arquivo ::= SEQUENCE
{
  TipoConteudo,
  outros      AtributosArquivo,
  conteudo    ANY }

```

- O tipo **SET** é usado para representar uma coleção de variáveis cujo número é conhecido e modesto e cuja ordem não é significativa. Neste caso, cada variável deve ser rotulada neste contexto. Exemplo:

```
NomeUsuario ::= SET
{
  nomePessoal      [0] IMPLICIT ISO646String,
  nomeOrganização [1] IMPLICIT ISO646String,
  nomePaís         [2] IMPLICIT ISO646String }

```

A palavra **OPTIONAL** pode ser colocada após uma variável do conjunto cujo aparecimento é opcional. Se os membros de um tipo SET são em número fixo, também deve ser atribuído um nome a cada membro cujo fim não é evidente de seu tipo. Exemplo:

```
AtributosArquivo ::= SET
{
  proprietario      [0] IMPLICIT NomeUsuario,
  tamanhoDoConteudoEmOctetos [1] IMPLICIT INTEGER,
  controleAcesso     [2] IMPLICIT ControloAcesso }

```

- O tipo **SET OF** pode ser usado para representar uma coleção de variáveis cujos tipos são os mesmos e cuja ordem não é significativa. Ex:

```
PalavrasChave ::= SET OF ISO646String --em qualquer ordem
```

- O tipo rotulado é usado para definir um tipo de dados de uso geral, independente de aplicação que deve ser distinguível de qualquer outro, por meio de sua representação. Exemplo:

```
ChaveCriptografia ::= [UNIVERSAL 24] IMPLICIT OCTET
STRING -- sete octetos
```

Também podem ser usados tipos rotulados para definir um tipo de dados de uso amplo e disperso, num particular contexto de apresentação e que deve ser distinguível (por meio de sua representação) de todos os outros tipos de dados usados naquele contexto de apresentação. Exemplo:

```
NomeArquivo ::= [APPLICATION 8] IMPLICIT SEQUENCE
{
  nomeDiretorio ISO646String,
  nomeRelativoArquivoDiretorio ISO646String }

```

Os rótulos específicos de um contexto são usados para distinguir os membros de um conjunto. Os rótulos numéricos devem começar de zero se sua única meta é distinguir os tipos. Exemplo:

```

RegistroCliente ::= SET
{nome           [0] IMPLICIT ISO646String,
 enderecoPostal [1] IMPLICIT ISO646String,
 numeroConta    [2] IMPLICIT INTEGER,
 debito         [3] IMPLICIT INTEGER --em centavos-- }

```

Quando um particular membro do conjunto recebeu um rótulo a nível de

aplicação não necessita receber rótulo específico de contexto.Ex:

```

RegistroProduto ::= SET
{
    CodigoUniforme,
    descrição      [0] IMPLICIT ISO646String,
    numeroEstoque  [1] IMPLICIT INTEGER }
CodigoUniforme ::= [APPLICATION 13] IMPLICIT INTEGER

```

Rotulação específica de contexto também é usada para distinguir as alternativas de uma escolha (CHOICE). Rótulos numéricos começam em zero se seu único propósito é distinguir os diversos tipos. Exemplo:

```

AtributoDeCliente ::= CHOICE
{nome           [0] IMPLICIT ISO646String,
 enderecoPostal [1] IMPLICIT ISO646String,
 numeroDaConta  [2] IMPLICIT INTEGER,
 debito         [3] IMPLICIT INTEGER -- em centavos--}

```

Os rótulos de uso privativo são usados para rotular tipos de dados que são de uso apenas no escopo de uma particular organização ou país e que devem ser distinguíveis de todos os demais tipos de dados usados por aquela organização ou país. Exemplo:

```
NumeroCartaoAcme ::= [PRIVATE 2] IMPLICIT INTEGER
```

- O tipo **CHOICE** (seleção) é usado para representar uma variável que é selecionada dentre uma coleção de variáveis cujo número é conhecido e modesto. Cada variável do conjunto deve ser identificada por um rótulo específico do contexto. Exemplo:

```

IdentificadorArquivo ::= CHOICE
{ nomeRelativo  [0] IMPLICIT ISO646String, -- nome do arquivo
  nomeAbsoluto  [1] IMPLICIT ISO646String, -- nome do arquivo e do
                                              directorio que o contem<
  numeroSerial  [2] IMPLICIT INTEGER
  --identificador atribuido pelo sistema }

```

O tipo seleção é usado para representar uma variável cujo tipo é um dentre alguma particular alternativa de um **CHOICE** previamente definido.

Exemplo: a definição seguinte é possível

```

AtributosCorrentes ::= SEQUENCE
{ data-ultimo-uso  AtributoArquivo,
  nome-arquivo     AtributoArquivo }

```

se existir a seguinte definição:

```

AtributoArquivo ::= CHOICE
{ data-ultimo-uso  INTEGER,
  nome-arquivo     ISO646String }

```

- O tipo **ANY** é usado para modelar uma variável cujo tipo não é especificado ou é especificado em outro ponto usando

ASN.1.Exemplo:

ConteudoMensagem ::= ANY

-- um elemento de dados cujo tipo e especificado na norma XXX

- O tipo **EXTERNAL** é usado para referenciar uma variável cujo tipo não é especificado ou que é especificado em outro local usando ASN.1 .

[Próxima](#)