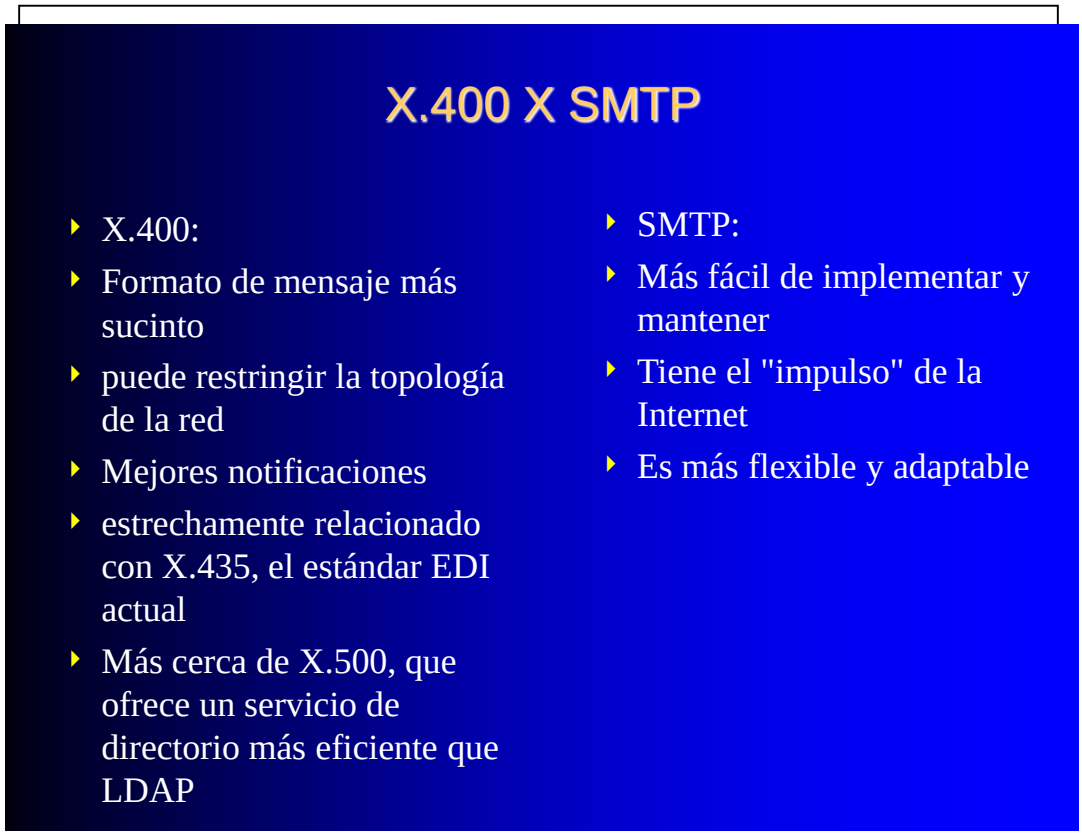
Taller AMHS 2

2023

Francisco Almeida – Oficial CNS SAM



Comparaciones entre sistemas X.400 y SMTP

Vale la pena mencionar la inevitable comparación que se hace entre las dos infraestructuras de redes de uso más amplio: el SMTP de Internet y el X.400 / MOTIS. Es cierto que el servicio de mensajería por Internet se considera más ampliamente difundido en términos de usuarios conectados. Sin embargo, aquellos usuarios y aplicaciones que tienen un aspecto de "misión crítica", que incluye la gestión del tráfico aéreo, generalmente tienden hacia el uso de MHS. John Rothon enumera algunas ventajas técnicas de X.400/MOTIS sobre SMTP [Rothon97]:

- Ofrece un formato de mensaje sucinto;

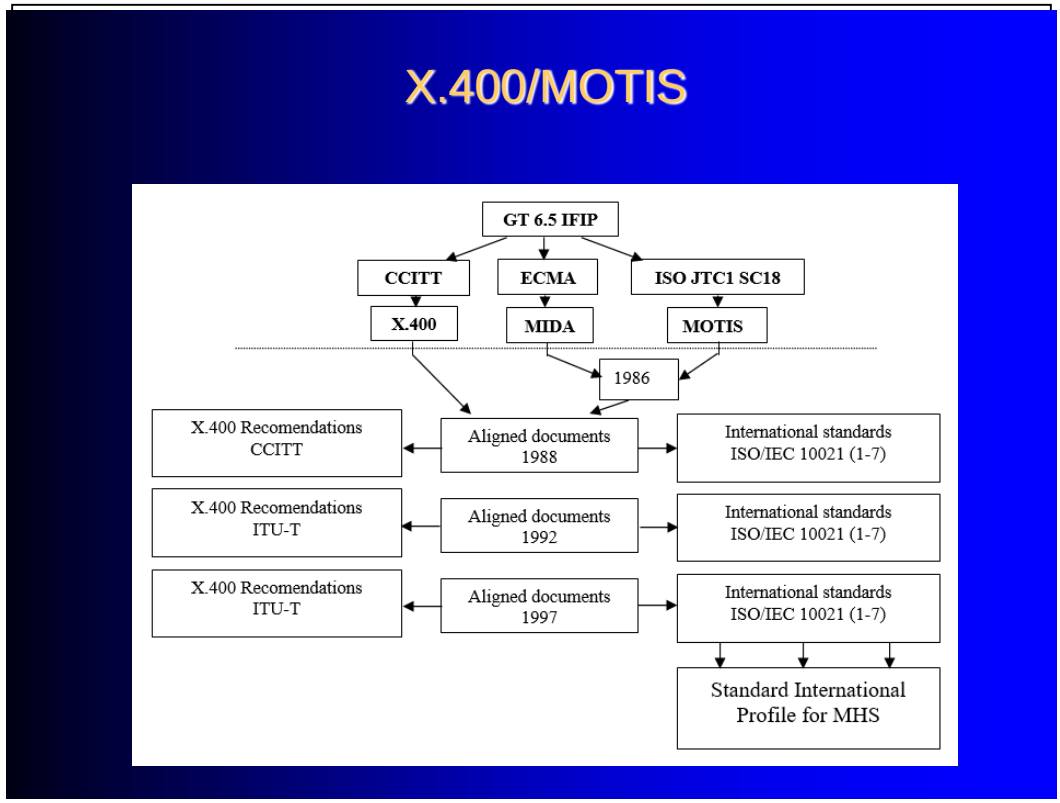
- Permite restringir la topología del *backbone* de mensajes;

- Ofrece mejores notificaciones;

- Está estrechamente relacionado con X.435, que es el estándar EDI actual; y,

- X.400 está más estrechamente relacionado con X.500, que ofrece un servicio de directorio más completo que el LDAP.

En una encuesta realizada por la Electronic Messaging Association - EMA en 1997, el setenta y cuatro por ciento de los gerentes y administradores de sistemas entrevistados indicaron una preferencia por productos basados en las recomendaciones X.400 para lograr mejores niveles de calidad para su negocio [Rubenstein98].



El desarrollo del Estándar X.400

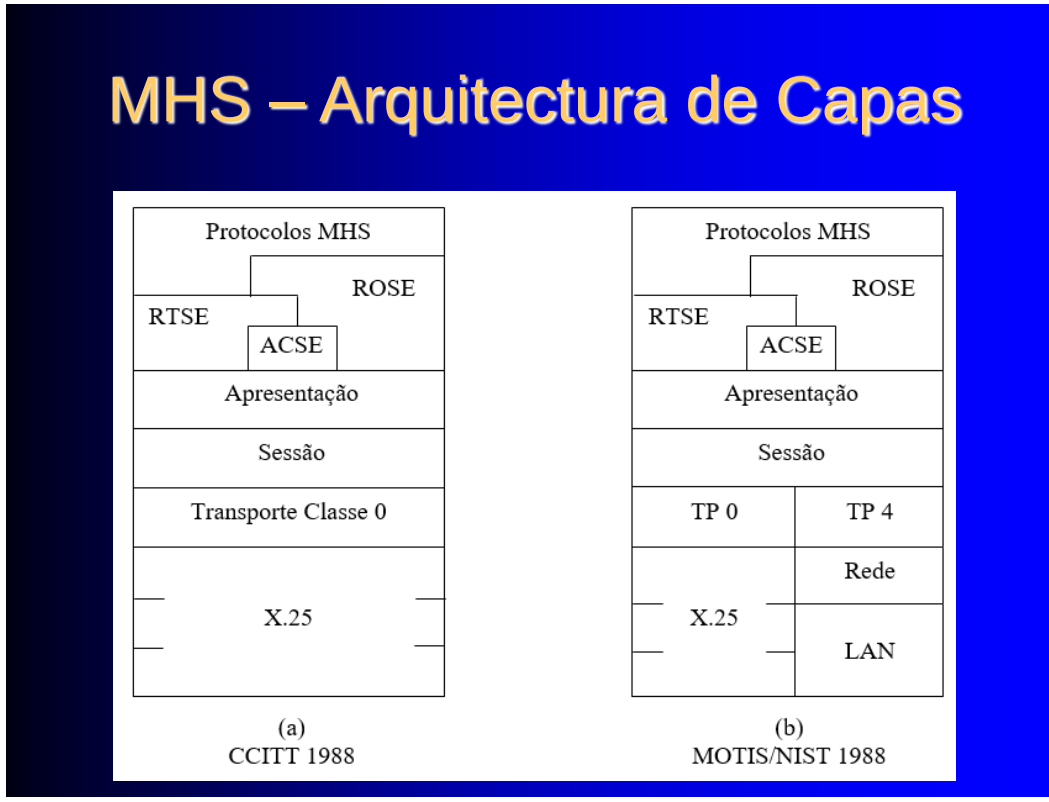
En 1975, la Federación Internacional de Procesamiento de la Información – IFIP comenzó el desarrollo de la definición de un sistema de mensajes generalizado, a través del Grupo de Trabajo 6.5. El objetivo era desarrollar los requisitos para los sistemas de mensajería basados en computadora (CBMS). Otras organizaciones de normalización han adoptado la idea y el modelo del IFIP. En 1981, el CCITT asumió el trabajo que culminó con la ratificación, en 1984, de la serie X.400 de Recomendaciones. En 1986, ISO decidió suspender el Borrador de Estándar Internacional y cooperar con el CCITT para producir un documento conjunto, ratificado en 1986.

La Asociación Europea de Fabricantes de Computadoras – ECMA que ya había definido la Aplicación Distribuida de Intercambio de Mensajes – MIDA, también se unió al esfuerzo. El resultado de este esfuerzo combinado fue el texto conjunto ISO/IEC/CCITT/ECMA [X.400], del cual se derivaron la serie ISO/IEC 10021 de Normas Internacionales y el Libro Azul de 1988 CCITT de Recomendaciones X.400.

En 1990 y 1992, hubo pocos cambios en la Recomendación X.400 existente, pero se definieron dos nuevos servicios, uno para el intercambio electrónico de datos – EDI manejado en las series F.435 y X.435 y otro para la mensajería vocal, con las series F.440 y X.440.

La modalidad de trabajo conjunto de normalización se mantuvo en las versiones de las normas de 1992, 1997 y 1999. Posteriormente, ISO/IEC ha desarrollado los Estándares de Especificación de Uso de Servicios de Directorio (X.500).

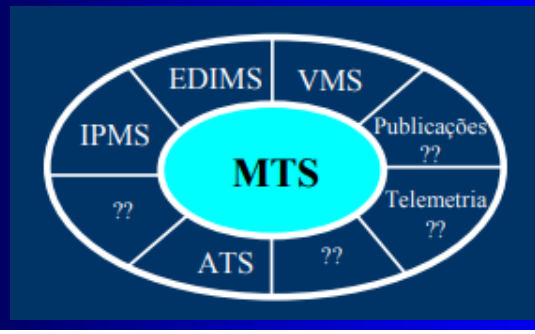
Los estándares MHS están conformados por funcionalidades y no todas las funciones son necesarias para todos los usuarios. En realidad, MHS puede ser compatible con una variedad de diferentes capas de protocolos de comunicación. Esto plantea la posibilidad de que diferentes proveedores de sistemas puedan desarrollar diferentes subconjuntos de funcionalidades y posiblemente implementar diferentes capas de protocolo.



La figura (a) muestra la versión de 1988 del antiguo CCITT, que utiliza X.25 y la clase de transporte 0 en las capas inferiores. La figura (b) muestra la versión ISO 1988 MOTIS, que también puede admitir la clase de transporte 4 y una variedad de tecnologías LAN en las capas bajas.

Servicios MHS

- Los estandares MHS incluyen especificaciones de un número de servicios para los usuarios:
  - Sevicio de Transferencia de Mensajes (MTS)
  - Servicio de Mensaje Interpersonal (IPM)
  - Servico de Mensaje de Intercambio de Datos (EDI)
  - Otros servicios



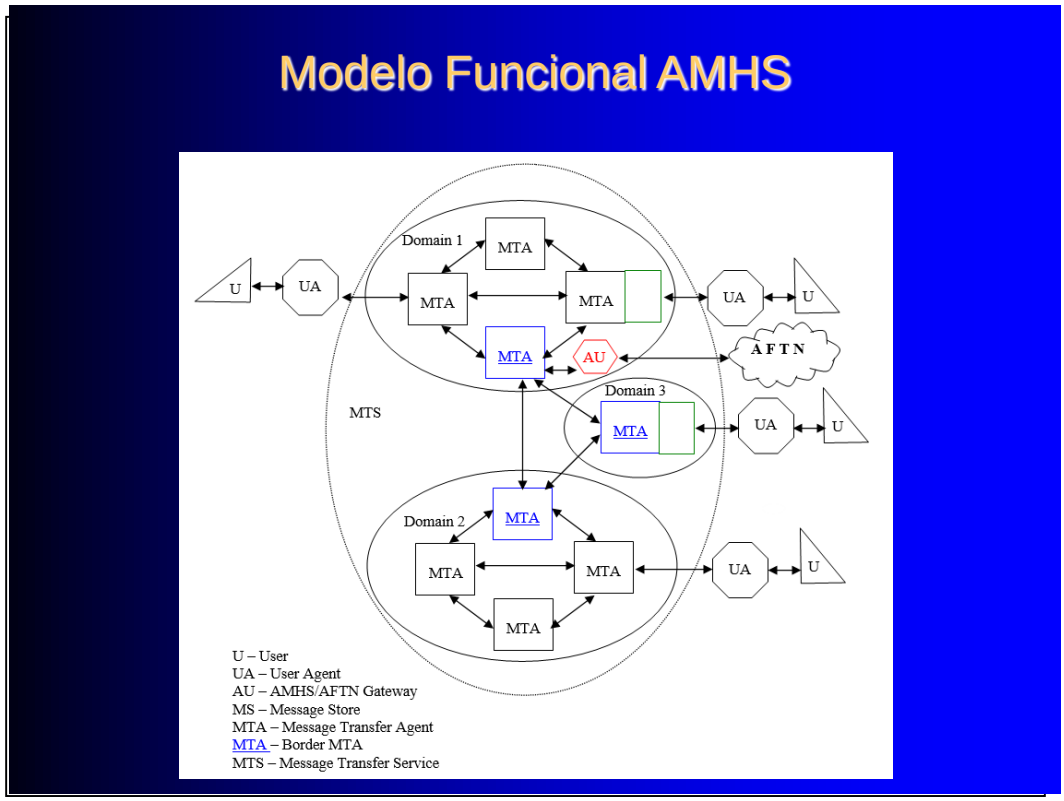
Se han especificado varios servicios en los estándares MHS, que son: Servicio de transferencia de mensajes, Servicio de mensajería interpersonal (IPMS), Servicio de mensajería de intercambio electrónico de datos (EDIMS) y Servicio de mensajería de voz (VMS).

El MTS es fundamental para todos los demás servicios. Es un servicio de entrega postal electrónica basado en los mecanismos de almacenamiento y envío y de direcciones múltiples. Principalmente, interpreta y actúa sobre la información contenida en el sobre (Header). Por lo general, no procesa el contenido a menos que se invoque la función especial de "conversión de contenido". Por lo general, es transparente al contenido del mensaje y, por lo tanto, permite que pase cualquier tipo de codificación de datos. Sus usuarios pueden ser personas o máquinas (computadoras).

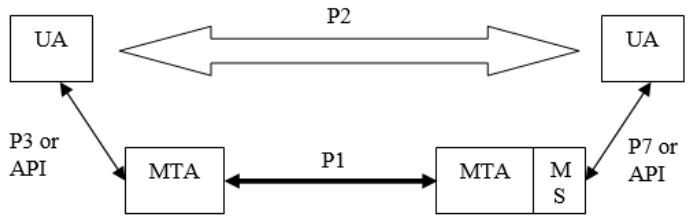
IPMS utiliza MTS para transferir mensajes entre usuarios. Ofrece comunicación entre personas en el estilo de correo electrónico, carta personal y memorándum. Especifica un encabezado predeterminado que permite al remitente indicar al destinatario las acciones que se deben realizar para procesar el mensaje. El mensaje puede constar de una o varias partes del cuerpo. También permite la transferencia de información compleja, documentos y archivos de los más variados tipos, tales como: hojas de cálculo, programas, gráficos, archivos, etc.

Outros serviços foram definidos, por exemplo: a condução de voz digitalizada (VMS),

mensagens de controle de tráfego aéreo (AMHS) e o armazena-e-envia de mensagens fax (COMFAX).



El conjunto de servidores de mensajes ATS, agentes de usuario y Gateways AFTN/AMHS se conoce como ATS Message Handling System (AMHS). El conjunto de protocolos implementados entre estos componentes se muestra en la figura abajo. Desde la perspectiva de la red ATN, las tres categorías de sistemas mencionadas anteriormente son Sistemas Finales ATN (ES).



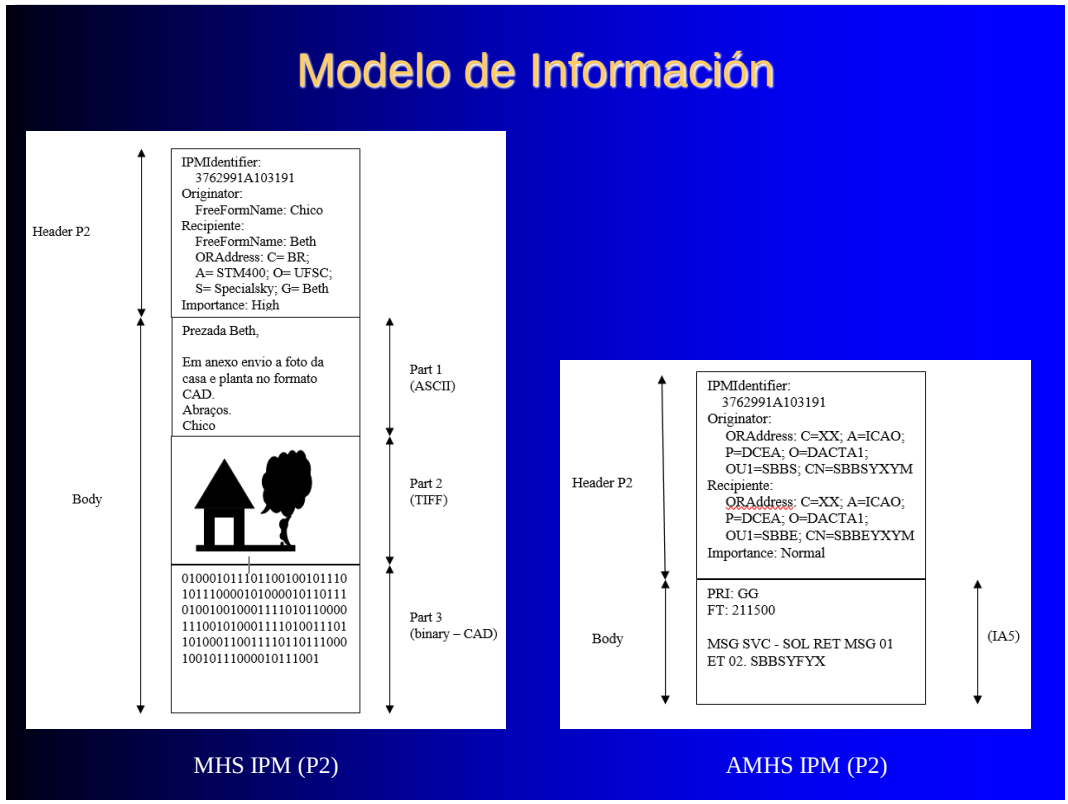
Cada dominio de administración debe estar interconectado con al menos otro dominio AMHS, que se denomina dominio adyacente. El concepto de dominios adyacentes no está relacionado con consideraciones geográficas, sino con la característica de comunicación directa entre los recursos pertenecientes a las organizaciones.

La comunicación entre dos dominios de administración de AMHS es siempre de MTA a MTA, es decir:

a) de un servidor de mensajería ATS a otro servidor de mensajería ATS;

b) de un servidor de mensajería ATS a un Gateway AFTN/AMHS, y viceversa; o

c) de un gateway AFTN/AMHS a otro gateway AFTN/AMHS.



En el servicio de mensajería básico ATS, el UA admite funciones adicionales específicas para la ATN para cumplir con los requisitos obligatorios de interconexión AFTN. Para ello, la UA utiliza una estructura de partes del cuerpo que reenvía elementos que son necesarios para la interconexión. Esta estructura comprende:

a) un elemento ATS_Message_Header, que reenvía parámetros AFTN que no tienen equivalentes directos en el estándar MHS; y

b) un elemento ATS_Message_Text, que reenvía el propio texto del mensaje.

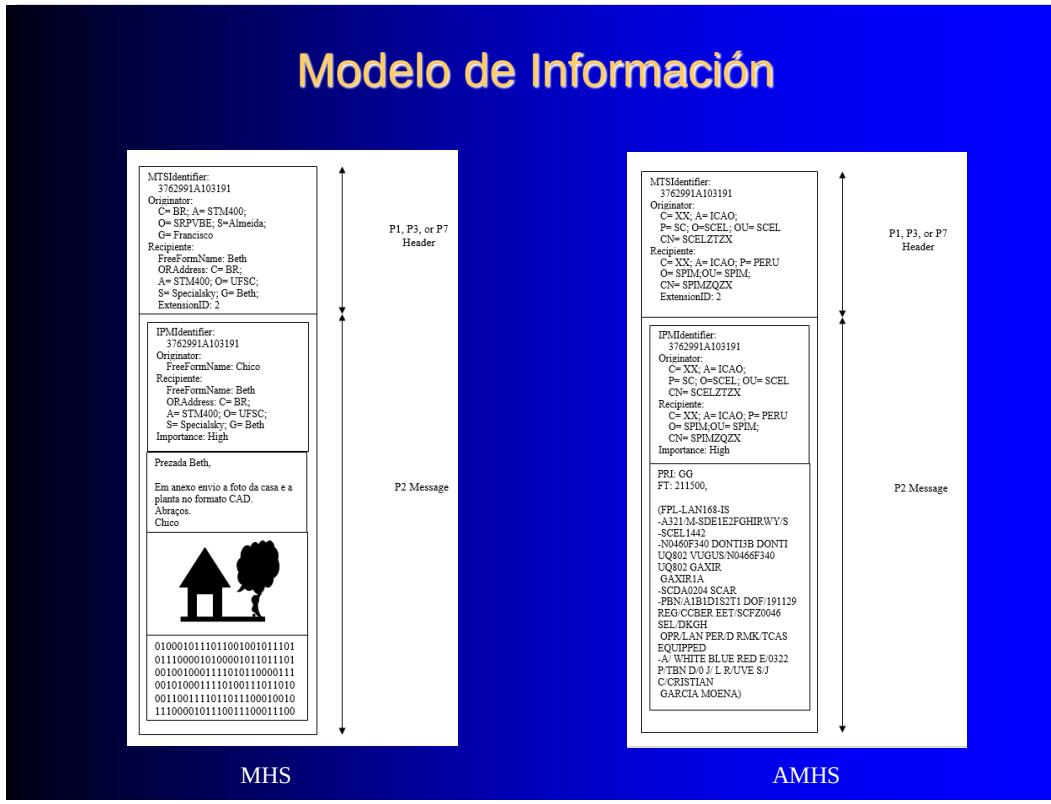
Los parámetros enviados a través del ATS_Message_Header son los siguientes:

a) Indicador de prioridad, que es enrutado en una estructura llamada **ATS_Message_Priority**;

b) Horario de llenado - FT, que se reenvía en una estructura llamada **ATS_Message_Filling_Time**; y

c) Información de encabezado opcional: OHI, que se reenvía en una estructura llamada **ATS_Message_Optional_Heading_Info**.

Para cumplir con los SARPS, una UA debe incluir la capacidad de soportar estos parámetros. Esto significa que la UA debe ser capaz de generar los elementos obligatorios, que son el **ATS_Message_Priority** y **ATS_Messagem_Filling_Time**, y opcionalmente poder generar el **ATS_Message_Opcional_Heading_Info**. Solo los mensajes dirigidos a AFTN, es decir, usuarios indirectos, requieren la generación de estos parámetros y la falta de estos parámetros provocará el rechazo del mensaje por parte del Gateway AFTN/AMHS.



El ATS Message Handling System (AMHS) es un subconjunto del MHS.

En el Servicio Básico, hay comunicación con usuarios indirectos (usuarios AFTN), siendo necesaria la inclusión de los elementos del **ATS_Message_Header** (**PRI:**, **FT:** y **OHI:**) en el cuerpo del IPM.

El AMHS utiliza esquemas particulares de direcciones.

Actualmente, dos esquemas son utilizados CAAS y XF

COMMON AMHS ADDRESS SCHEME (CAAS)

Una dirección con el formato CAAS es compuesto de los siguientes atributos:

C=XX (Pais=XX que fue atribuido para uso de los Estado Miembros de la OACI);

A=ICAO (Dominio administrativo de la OACI);

P=variable (Dominio privado de un Estado);

O=variable (Organizació en un Estado – normalmente local del MTA);

OU=variable (Unidade organizacional – normalmente un centro ATS); y

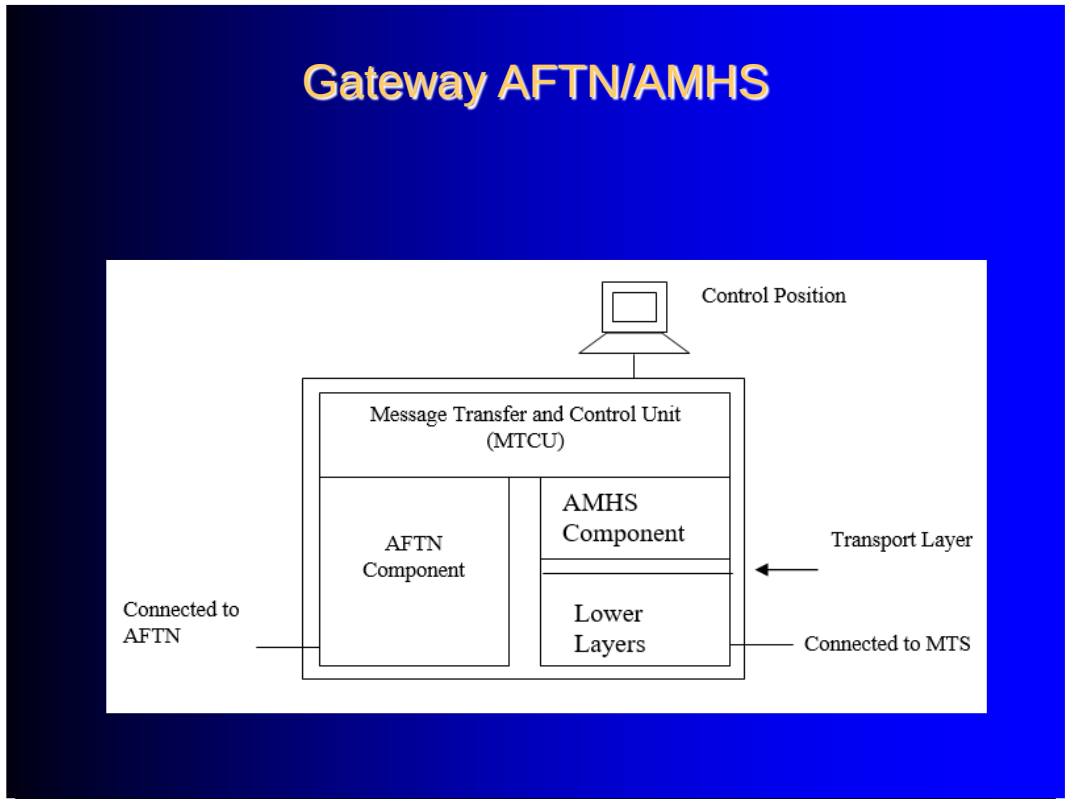
CN=variable (Common name – normalmente la antigua dirección AFTN).

Ejemplo: /C=XX/A=ICAO/P=SPAIN/O=LECS/OU=LEMG/CN=LEMGZPZX/

XF SCHEME

Una dirección XF tiene é atributo O=AFTN e el atributo OU es la dirección antigua AFTN del centro ATS (el atributo CN no es utilizado).

Ejemplo: /C=XX/A=ICAO/P=K/O=AFTN/OU=KATLZPZX/



Tres componentes principales definen el Gateway: el componente AFTN, el componente ATN y la unidad de transferencia y control de mensajes (UCTM).

COMPONENTE AFTN

El componente AFTN establece una conexión completa desde la puerta de enlace a un centro AFTN, con la capacidad de enviar y recibir mensajes. La puerta de enlace debe funcionar con un conjunto completo de funciones, apareciendo para el centro adyacente como se fuera una estación AFTN. Se debe asignar una dirección AFTN al componente AFTN.

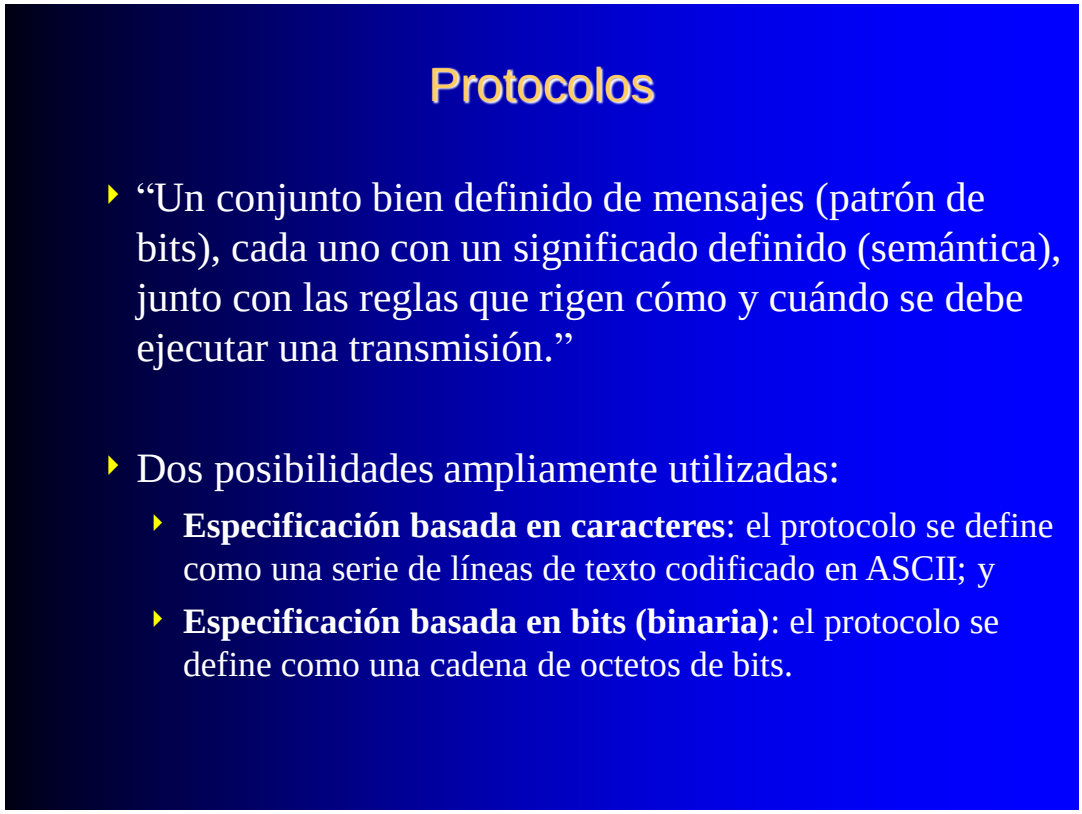
COMPONENTE AMHS

El componente ATN permite que la puerta de enlace funcione como un sistema final ATN. Equivalente al servidor de mensajería ATS, el componente ATN incorpora un MTA. Este MTA debe implementar el grupo funcional de lista de distribución según las especificaciones del servidor de mensajería ATS. Si lo desea, el dominio de administración puede implementar otros grupos de funcionalidad opcionales dentro del componente ATN de la puerta de enlace.

UNIDADE DE CONTROLE E TRANSFERÊNCIA DE MENSAGEM

En el gateway AFTN/AMHS, la unidad de transferencia y control de mensajes proporciona funciones a nivel de aplicación relacionadas con la unidad de acceso MHS - AU, que no forman parte del componente AFTN ni del componente ATN. Estas funciones conectan e integran los otros dos componentes y son esenciales para el funcionamiento de la pasarela

(Gateway).



Un protocolo se puede definir como:

“Un conjunto bien definido de mensajes (patrón de bits), cada uno con un significado definido (semántica), junto con reglas que rigen cómo y cuándo se debe ejecutar una transmisión.”.

Un protocolo rara vez existe solo. Por lo general, es parte de una pila de protocolos, en la que varias especificaciones distintas trabajan juntas para determinar el mensaje completo enviado por el remitente, con algunas partes de este mensaje destinadas a la acción de los nodos intermedios (sistemas) y otras partes dirigidas al sistema final remoto.

ESPECIFICACIÓN DE PROTOCOLOS

Los protocolos se pueden especificar de varias maneras. Dos posibilidades que son ampliamente utilizadas y que mantienen distinciones fundamentales entre ellas, son:

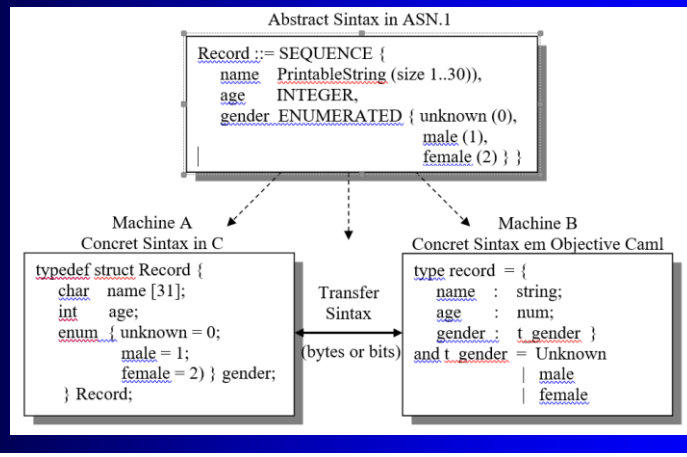
Especificación basada en caracteres: el protocolo se define como una serie de líneas de texto codificado en ASCII; y

Especificación basada en bits (binaria): el protocolo se define como una cadena de octetos de bits.

El segundo se conoce como el enfoque de sintaxis abstracta. Este es el enfoque utilizado por ASN.1, que tiene la ventaja de permitir al desarrollador producir especificaciones sin preocuparse por los detalles de codificación; y también permite el uso de herramientas para una fácil implementación de los protocolos especificados. Además, debido a que es independiente de la codificación, facilita la migración a codificaciones más avanzadas cuando sean desarrolladas.

Sintaxis

- Sintaxis Concreta
- Sintaxis Abstracta
- Sintaxis de Transferencia



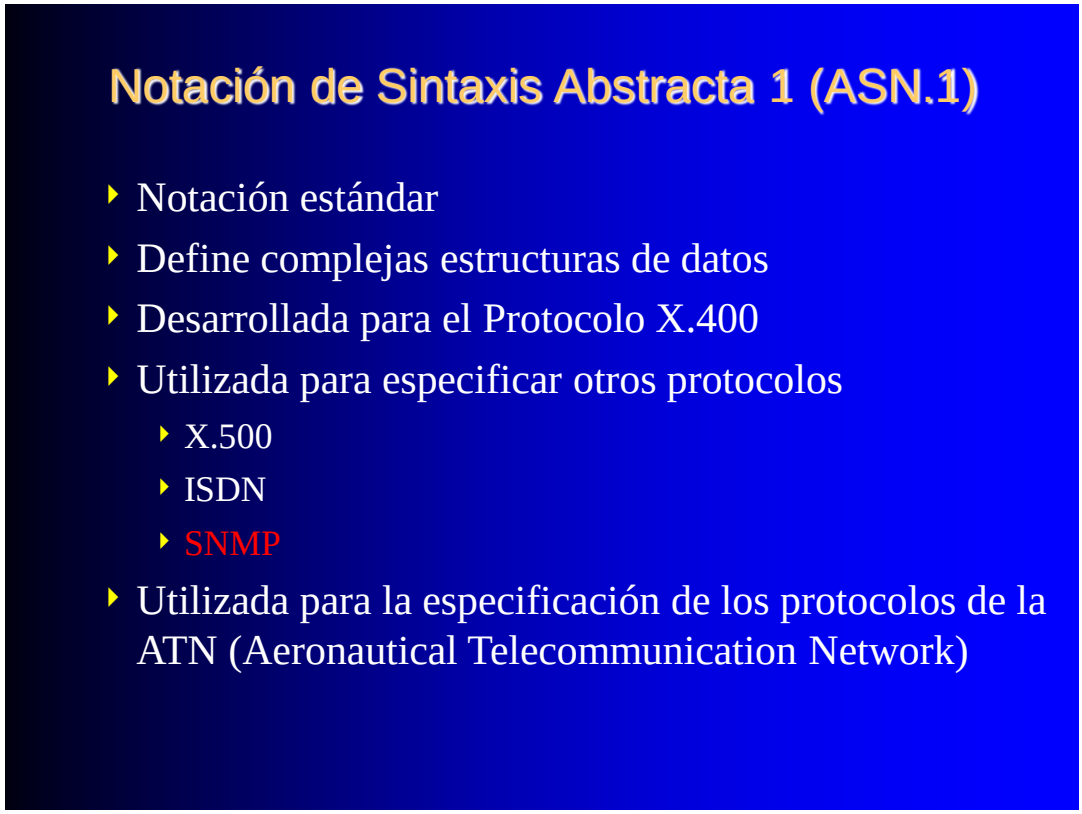
Sintaxis concreta: la sintaxis concreta es la representación de las estructuras de datos que se transferirán en un lenguaje de programación determinado. Es una sintaxis porque respeta las reglas léxicas y gramaticales (C, por ejemplo); Se llama concreto porque en realidad es manejado por la aplicación (implementado en el mismo lenguaje) y es compatible con las restricciones de la máquina (hardware).

Sintaxis abstracta: para ser independientes de las diversas sintaxis concretas (C, C++, Pascal, Fortran, etc.), las estructuras de datos a transmitir deben describirse sin conexión con el lenguaje de programación utilizado. Esta descripción también debe respetar las reglas léxicas y gramaticales de un "determinado lenguaje", pero debe permanecer independiente de los lenguajes de programación y nunca debe implementarse directamente en la máquina. Por estas razones, dicha descripción se denomina sintaxis abstracta, y ASN.1 es el "lenguaje" por el cual se denota esta sintaxis abstracta.

Sintaxis de Transferencia: los datos se reciben como una cadena de bytes o bits, que debe ser compatible con una sintaxis llamada sintaxis de transferencia, para que estas cadenas puedan ser reconocidas correctamente por la máquina..

Por supuesto, esta sintaxis de transferencia depende completamente de la sintaxis abstracta, ya que ensambla cómo se deben transmitir los datos, de acuerdo con la sintaxis abstracta. De hecho, la sintaxis de transferencia estructura y ordena los bytes que se enviarán a otra máquina. Pero a diferencia de la sintaxis abstracta es una cantidad física, y debido a esto, debe tener en cuenta el orden de los bytes, el peso de los bits, etc.

Se pueden asociar diferentes sintaxis de transferencia con una sola sintaxis abstracta. Esto es particularmente interesante cuando el flujo de datos (throughput) aumenta y se hace necesaria una codificación más compleja. En tal caso, es posible cambiar la sintaxis de transferencia sin cambiar la sintaxis abstracta.



La notación de sintaxis abstracta número uno (ASN.1) es una notación estándar para representar estructuras de datos de una manera independiente de la implementación. Fue desarrollado originalmente por CCITT para definir las complejas estructuras de datos empleadas en los protocolos X.400, y posteriormente fue adoptado por los escritores de estándares ISO para definir las unidades de datos de protocolo utilizadas en los protocolos de aplicación OSI. Dentro de la ATN, ASN. 1 se ha utilizado para la definición formal de mensajes de aplicación y para la definición de las unidades de datos de protocolo de capa superior utilizadas por las mejoras de eficiencia de la capa superior. Las normas ASN.1 son ISO/IEC 8824-1 y UIT-T Rec. X.208.

Tipos primitivos

ASN.1 se basa en varios tipos de datos primitivos y mecanismos para construir, a partir de estas primitivas, tipos estructurados para adaptarse a los requisitos de la aplicación. ASN.1 también permite que los tipos estructurados se definan recursivamente para construir tipos estructurados adicionales y cada vez más complejos. Los tipos se nombran y definen mediante instrucciones de asignación de tipos, y las instrucciones de asignación de valor permiten asignar valores a una instancia específica de cualquier tipo primitivo o a un tipo estructurado de complejidad arbitraria.

Tipos definidos

Para cualquiera de los tipos primitivos de ASN.1, los tipos simples pueden definirse utilizando instrucciones de asignación de tipos para, en efecto, cambiar el nombre de un tipo primitivo y, opcionalmente, definir el dominio de valores que ese tipo puede tomar.

Tipos estructurados

Los mecanismos ASN.1 para definir tipos estructurados son listas, repeticiones y alternativas.

Las palabras clave SEQUENCE y SET se utilizan para definir tipos estructurados que consisten en un conjunto de otros tipos, ya sean simples o estructurados por sí mismos. La diferencia entre los dos es que en el tipo SET, no hay importancia para el orden en que se enumeran los subtipos, mientras que en el tipo SEQUENCE, los valores siempre se asignarán en el orden en que se definen los subtipos.

Clases de tipos de la ASN.1

- Universal
- Amplia aplicación
- Contexto específico
- Privada

Cada tipo se distingue por una etiqueta, que especifica la clase del tipo e identifica el tipo particularmente. Por ejemplo, UNIVERSAL 4 hace referencia a OctetString, que es de la clase UNIVERSAL y tiene ID 4 dentro de esa clase.

ASN.1 como se describe en la Recomendación X.409 tiene cuatro clases de tipos:

Universal: tipos de utilidad general, independientemente de la aplicación y mecanismos de construcción; estos se definen en la norma y se enumeran en la tabla a continuación.

TAG	TYPE	TAG	TYPE
UNIVERSAL 1	Boolean	UNIVERSAL 18	NumericString (Character String)
UNIVERSAL 2	Integer	UNIVERSAL 19	Printable String (Character String)
UNIVERSAL 3	BitString	UNIVERSAL 20	TeletexString (Character String)
UNIVERSAL 4	OctectString	UNIVERSAL 21	VideotexString (Character String)
UNIVERSAL 5	Null	UNIVERSAL 22	IA5String (Character String)
UNIVERSAL 6	Object Identifier	UNIVERSAL 23	UTCTime
UNIVERSAL 7	Object Descriptor	UNIVERSAL 24	Generalized Time
UNIVERSAL 8	External	UNIVERSAL 25	Graphic String (Character String)
UNIVERSAL 9-15	Reserved	UNIVERSAL 26	Visible String (Character String)
UNIVERSAL 16	Sequence and Sequence-of	UNIVERSAL 27	General String (Character String)
UNIVERSAL 17	Set and Set-of	UNIVERSAL 28	Reserved

Amplia aplicación: pertinentes para una aplicación concreta; estos se definen en otras normas.

Contexto específico: también pertinentes para una aplicación concreta, pero aplicables a un contexto limitado.

Privada: Tipos definidos por el usuario que no están cubiertos por ninguna norma.

La sintaxis se utiliza tanto para definir tipos de datos como para especificar valores de datos. La mejor manera de describir la sintaxis es con un ejemplo.

Tomemos los datos de Registro de Personal de una empresa ficticia. Descritos sin la formalidad ASN.1 en la diapositiva.

Utilizando la ASN.1 para describir la estructura de datos, se quedaría:

PersonnelRecord ::= [APPLICATION 0] IMPLICIT SET {

Name,

title [0] IA5String,

EmployeeNumber,

dateOfHire [1] Date,

nameOfSpouse [2] Name,

[3] IMPLICIT SEQUENCE OF ChildInformation DEFAULT {}}

ChildInformation ::= SET {

Name,

dateOfBirth [0] Date}

Name ::= [APPLICATION 1] IMPLICIT SEQUENCE {

givenName IA5String,

initial IA5String,

FamilyName IA5String}

EmployeeNumber ::= [APPLICATION 2] IMPLICIT INTEGER

Date ::= [APPLICATION 3] IMPLICIT IA5String -- YYYYMMDD

La configuración **EmployeeNumber ::= {APPLICATION 2] IMPLICIT INTEGER** utiliza el tipo INTEGER nativo (built-in), pero el usuario ha elegido asignar una nueva etiqueta al tipo. El uso del término [APLICACIÓN 2] proporciona una etiqueta (clase e ID) para este nuevo tipo. La designación IMPLÍCITA está relacionada con la representación de los valores en la sintaxis de transferencia. Con ese término presente, los valores de este tipo se codificarán solo con la etiqueta APPLICATION 2. Si la designación no estaba presente, los valores serían codificados por las etiquetas APPLICATION y UNIVERSAL. El uso de la opción implícita da como resultado una representación más compacta.

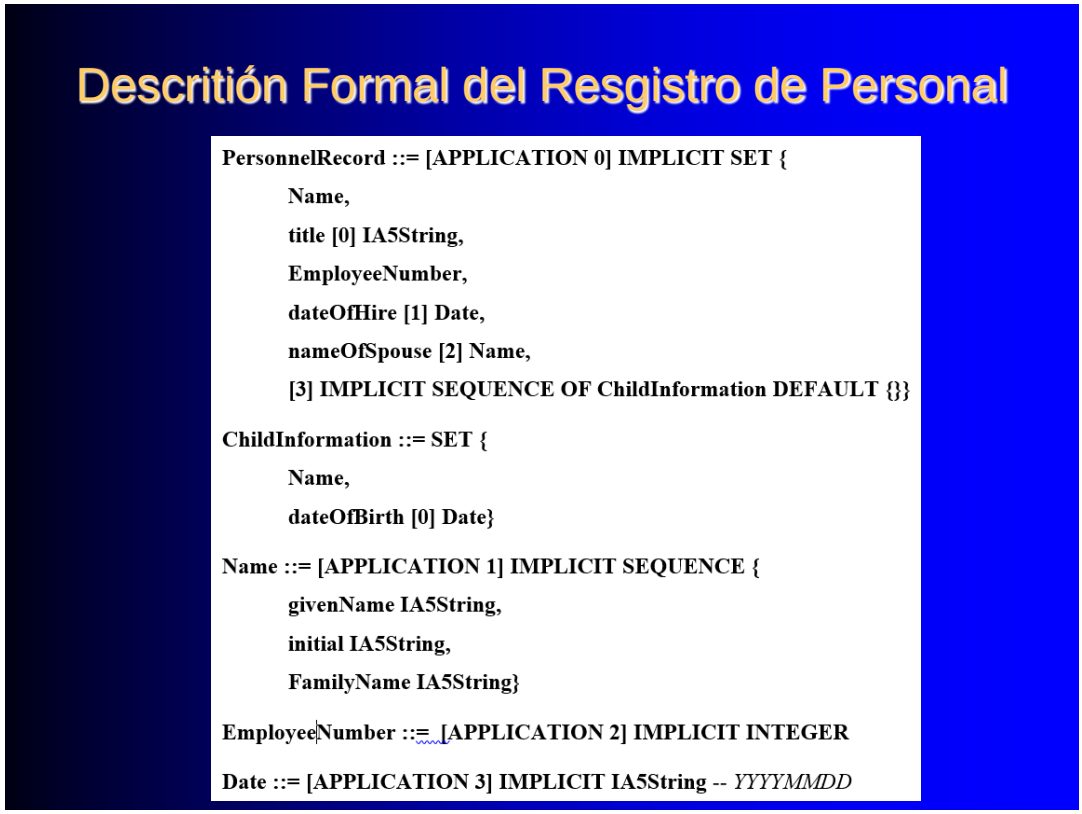
La definición del tipo **Date** es similar a la del tipo **EmployeeNumber**. En este caso, el tipo es una cadena que consiste en el juego de caracteres conocido como IA5. El guion doble indica que el resto de la línea es comentario; el formato de los valores de tipo Date no se comprobará más allá de determinar que el valor es una cadena IA5.

Los otros tipos son más complejos. Considere la posibilidad de establecer el tipo **Name**. El tipo básico de **Name** es una **SEQUENCE**, que define una serie ordenada de otros tipos. Una secuencia es análoga a la estructura de archivos que se encuentra en muchos lenguajes de programación, como COBOL, por ejemplo. El principio y el final de la definición de la secuencia están marcados por llaves. Una secuencia consta de una serie de elementos, que especifican un tipo. Se permiten tres formas de SEQUENCE:

- un número variable de elementos, todos de un tipo (se utiliza la designación SECUENCIA DE);

- un número fijo de elementos, posiblemente de más de un tipo; y

- un número fijo de elementos, algunos de los cuales son opcionales; donde todos los elementos, incluidos los opcionales, deben ser de diferentes tipos.

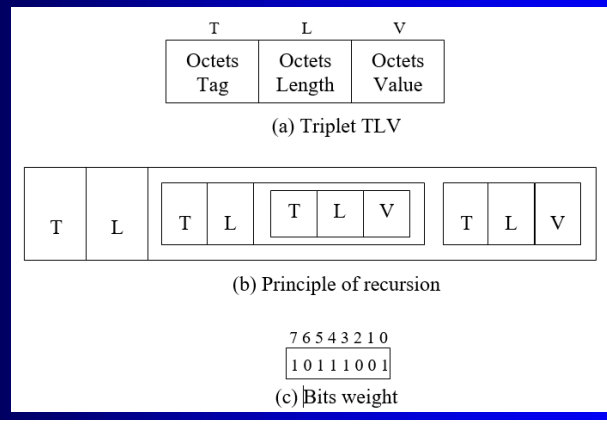


Ahora considere la definición de **ChildInformation**, que es un tipo **SET** (conjunto). Un SET es similar a una SEQUENCE, excepto que el orden de los elementos no es significativo. Los elementos se pueden organizar en cualquier orden cuando están codificados en una representación específica. Al igual que en Sequence, se permiten las tres formas mencionadas anteriormente. En el ejemplo, Set contiene dos elementos. El primero es el tipo de datos **Name**, que se define en la secuencia ya discutida. Este es un ejemplo de la capacidad recursiva para definir otros tipos con tipos. Tenga en cuenta que no se le da ningún nombre al primer elemento, pero al segundo elemento se le da el nombre **dateOfBirth**. El segundo elemento es el tipo de datos **Date** definido en otra parte. Este tipo se utiliza en dos contextos diferentes, aquí en la definición de **PersonnelRecord**. En cada contexto, el tipo de datos recibe un nombre y una etiqueta específica del contexto [0] y [1], respectivamente.

Finalmente, la estructura general, **PersonnelRecord** se define como un **SET** con cinco elementos. Asociado con el último elemento hay un valor predeterminado de una cadena nula, que se utilizará si no se proporciona ningún valor.

Reglas Básicas de Codificación

La sintaxis de transferencia de Basic Encoding Rules (BER) siempre tiene el formato de un triplet TLV (Type, Lenght y Value) también expresado como <Tag, Length, Value>. Todos los campos T, L y V son series de octetos. El campo V puede tener en sí mismo otra triplet T, L y V, se es un tipo construido, ver figura (b).

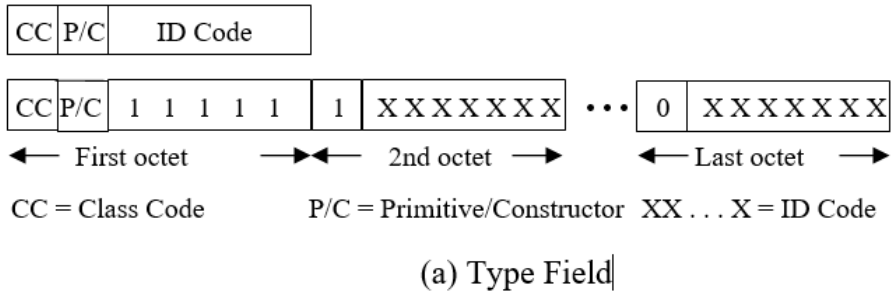


El campo Type codifica la etiqueta (Tag – clase y número) del valor del tipo de datos. Los dos primeros bits indican a cuál de las cuatro clases (universal, aplicación amplia, contexto específico o privado) pertenecen los tipos. El siguiente bit indica si la forma del tipo es primitiva o de construcción.

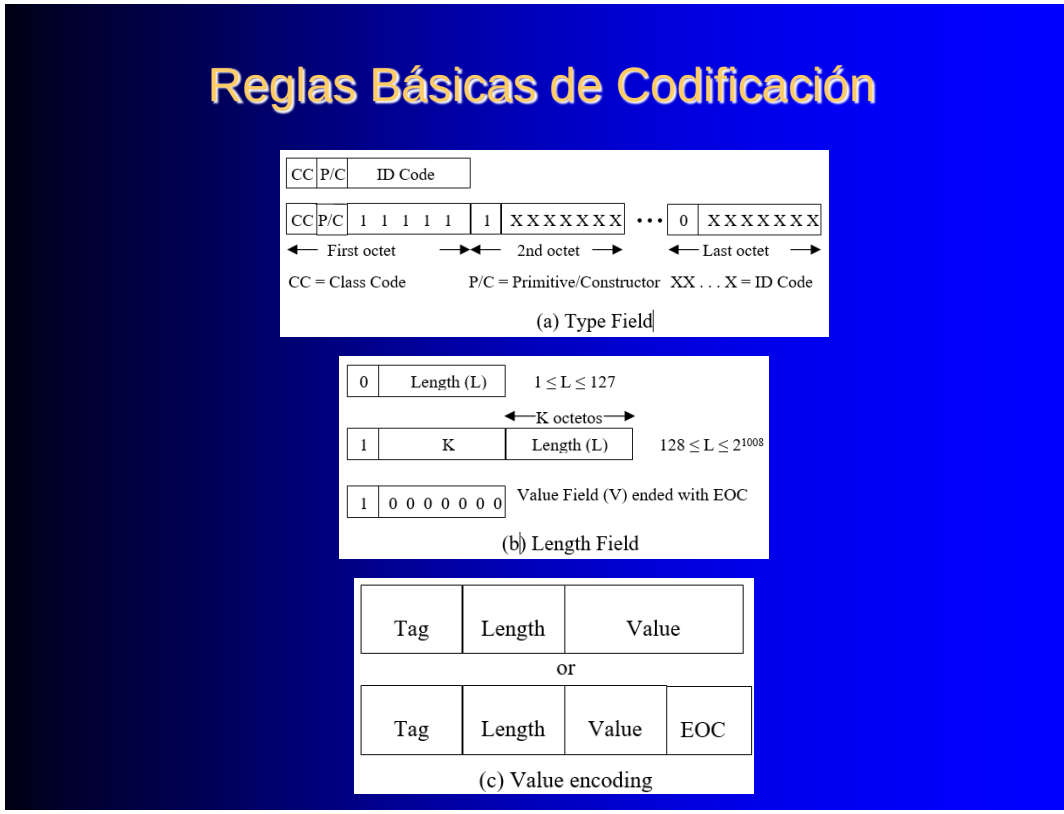
- Primitivo (básico): El contenido del campo Valor representa directamente el valor; y

- Constructor: El contenido del campo Valor es la codificación completa de uno o más valores de datos. Se utiliza para tipos como Sequence y Set.

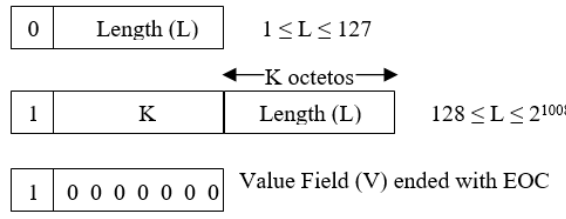
Los cinco bits restantes del campo Type pueden codificar un código de ID numérico que distingue un tipo de datos de otro tipo dentro de la clase designada. Para las etiquetas cuyo número es mayor que 31, estos cinco bits contienen el valor binario 11111 y el ID está contenido en los últimos siete bits de uno o más octetos adicionales. El primer bit de cada octeto adicional indica si se trata del último octeto adicional en el campo Type.



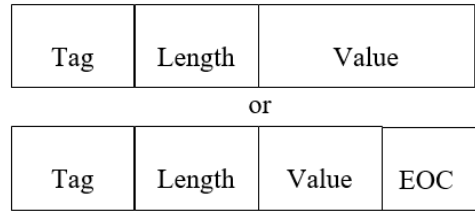
(a) Type Field



El campo Length especifica el tamaño en octetos del campo Value. Para L menor que 128, el campo L consiste en un solo octeto que comienza con cero. Si la longitud del campo Value no se conoce inmediatamente, pero es mayor que 127, el primer octeto del campo Length contiene un código de siete bits que especifica el tamaño del campo L. Si el tamaño del campo Value no se conoce inmediatamente, el campo L toma un valor binario de 10000000 y el campo Value termina con un marcador de fin de contenido que consta de 16 bits cero (End of Content – EOC).



(b) Length Field



(c) Value encoding