Zürcher Hochschule Winterthur

Abstract Syntax Notation One ASN.1

Dr. Andreas Steffen

ASN.1 – Abstract Syntax Notation One

- Standards and applications using ASN.1
- Abstract syntax and transfer syntax
- Simple types – basic types, character string types
- Date and time types
- Type definitions
- Subtype definitions
- Value Assignments
- Object identifier type (OID)
- Structured types – sequence, set
- Context-specific tags

BER – Basic Encoding Rules

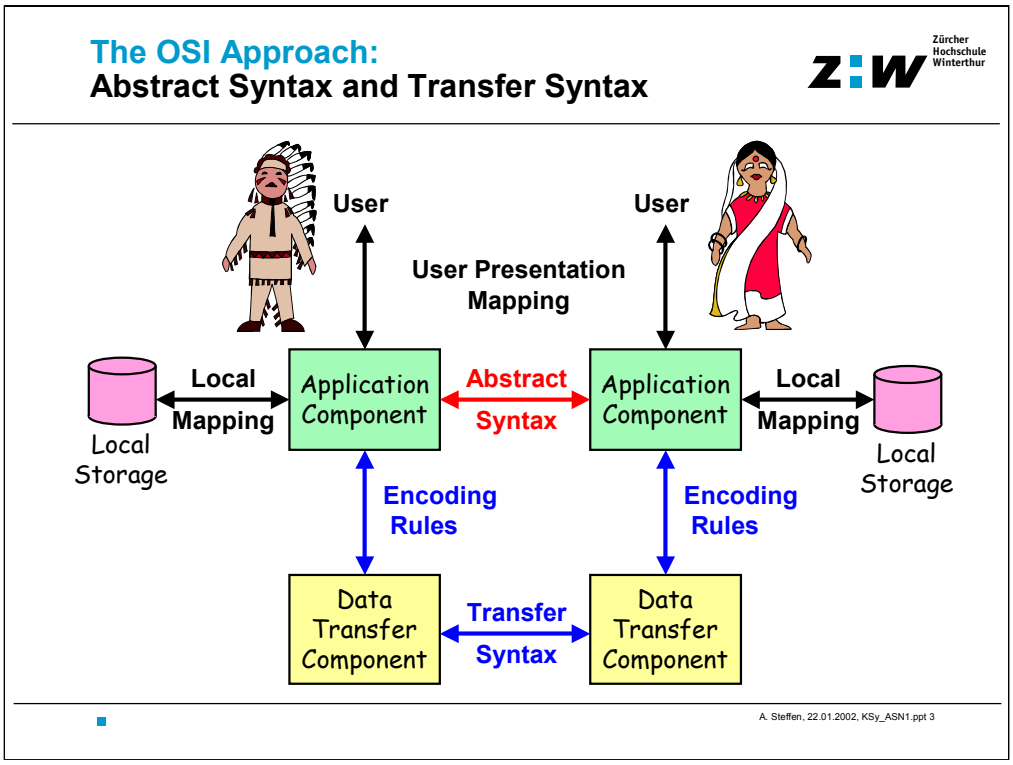
- Type–length–value rule
- Type tags
- Encoding of identifier field
- Encoding of tag classes
- Encoding of tag numbers
- Encoding of length field
- Encoding of integers
- Encoding of sequences
- Encoding of object identifiers
- Alternative encoding rules – PER, DER, CER

Standards and Applications using ASN.1

Zürcher Hochschule Winterthur

- X.400 Message Handling System / X.500 Directory Services
  - LDAP based Directories / X.509 Digital Certificates
- RSA Public Key Cryptography Standards
  - Storage and Transmission of Keys / Certificates (PKCS #12)
- Secure Electronic Transaction (SET)
  - Visa / Mastercard
- H.323 / T.120 Multimedia Communication Standards
  - Definition and Exchange of Multimedia Terminal Capabilities
- Telecommunications Management Network (TMN)
  - Performance / Fault and Configuration Management Information
- Unicode Worldwide Character Standard
  - Universal 16-bit Character Set comprising all written languages
- Simple Network Transfer Protocol (SNMP)
  - Management Information Base (MIB) / SNMP PDUs

Source: OSS Nokalva, http://www.oss.com/asn1/usage.html



Abstract Syntax

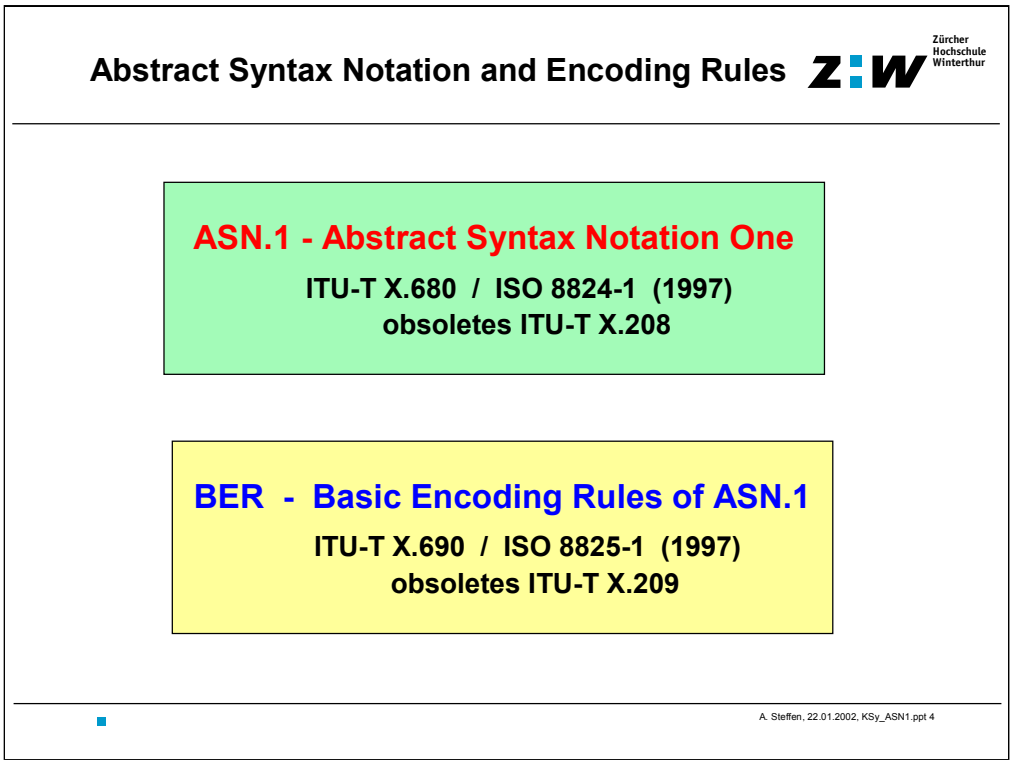
- An abstract syntax provides a set of formal rules for describing the structure of objects independent of both the user presentation and the machine-specific encoding techniques used for storage and transmission of the objects.

Transfer Syntax

- The transfer syntax is the actual representation of data as it is transmitted over a network. Often data is represented by a stream of octets.

Encoding Rules

- Encoding rules define the mapping of the abstract syntax used to represent data objects into the transfer syntax used to transmit the objects over a physical transmission channel.



Abstract Syntax Notation One

- ASN.1 is especially well suited for the representation of the complex, variable and extensible data structures used in modern communications applications.

Basic Encoding Rules of ASN.1

- BER encodes ASN.1 objects into a series of octet strings that can be transmitted over an octet oriented communications channel

ASN.1 Simple Types I

Zürcher Hochschule Winterthur

- **Basic Types**
  - BOOLEAN
  - INTEGER
  - ENUMERATED
  - REAL
  - BIT STRING
  - OCTET STRING
- **Character String Types (various subsets of ISO 10646-1)**
  - NumericString (0-9,<space>)
  - PrintableString (0-9,A-Z,a-z,<space>,<special>)
  - VisibleString
  - GraphicString
  - TeletexString
  - UTF8String
  - IA5String

ASN.1 Simple Types II

Zürcher Hochschule Winterthur

- **Object Types**
  - `OBJECT IDENTIFIER`
  - `ObjectDescriptor`
- **Miscellaneous Types**
  - `NULL`
  - `UTCTime`
    `yymmdd hhmm[ss] <local offset from UTC>`
  - `GeneralizedTime`
    `yyyymmdd hhmm[ss] <local offset from UTC>`

NULL

- The NULL type denotes a null value.

OBJECT IDENTIFIER

- The OBJECT IDENTIFIER type denotes an object identifier, a sequence of integer components that identifies an object such as an algorithm, an attribute type, or perhaps a registration authority that defines other object identifiers. An OBJECT IDENTIFIER value can have any number of components, and components can generally have any nonnegative value.

UTCTime

- yy = 50..99 : 1950-1999
- yy = 00..49 : 2000-2049

GeneralizedTime

- Mandatory starting with the year 2050

ASN.1 Type Definitions

Zürcher Hochschule Winterthur

Syntax: `<type name> ::= <type>`

Examples:

```
Counter    ::= INTEGER

IpAddress  ::= OCTET STRING

Months     ::= ENUMERATED {january (1),
                           february (2),
                           march (3),
                           april (4),
                           may (5),
                           june (6),
                           july (7),
                           august (8),
                           september (9),
                           october (10),
                           november (11),
                           december (12)}
```

ASN.1 Subtype Definitions

Zürcher Hochschule Winterthur

Syntax: `<subtype name> ::= <type> ( <constraint> )`

Examples:

```
Counter     ::= INTEGER ( 0..4294967295 )
IpAddress   ::= OCTET STRING ( SIZE(4) )
Spring      ::= Months ( march | april | may )
Summer      ::= Months ( june | july | august )
SmallPrime  ::= INTEGER ( 2 | 3 | 5 | 7 | 11 )
ExportKey   ::= BIT STRING ( SIZE(40) )
```

ASN.1 Value Assignments I

Zürcher Hochschule Winterthur

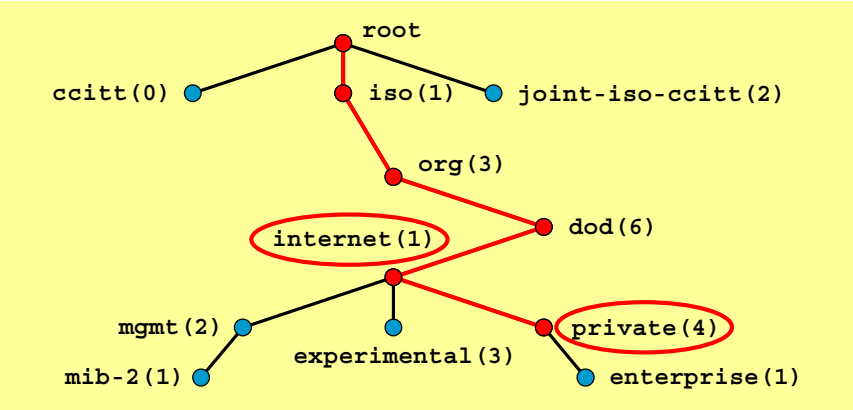
Syntax: `<value name> <type> ::= <value>`

Examples:

```
ipInReceives Counter     ::= 2450
ipRouteMask IpAddress    ::= 'FFFFFF00'H
currentMonth Months      ::= february
currentTime UTCTime      ::= "000204075015+0100"
givenName VisibleString  ::= "Andreas"
married BOOLEAN          ::= TRUE
faxMessage BIT STRING    ::= '01100001101'B
encryptionKey ExportKey  ::= 'A1B2C3D4E5'H
```

ASN.1 Value Assignments II

OBJECT IDENTIFIER




```
internet OBJECT IDENTIFIER ::=
                          { iso(1) org(3) dod(6) 1 }
private  OBJECT IDENTIFIER ::= { internet 4 }
```


Hierarchical Tree Structure

- ASN.1 object identifiers are organized in a hierarchical tree structure to make it possible to give any object a unique global identifier.

SNMP Objects

- The official SNMP objects defined by various RFCs are attached below the node
  `{ iso(1) org(3) dod( 6) internet(2) mgmt(1) }`
- Enterprise specific SNMP objects are attached below the node
  `{ iso(1) org(3) dod( 6) internet(2) private(4) enterprise(1) }`

SNMP OIDs of some well-known enterprises

- IBM: { enterprises 2 }
- Cisco: { enterprises 9 }
- Hewlett-Packard: { enterprises 11 }
- Sun Microsystems: { enterprises 42 }
- Microsoft: { enterprises 311 }
- Intel: { enterprises 343 }

Security OIDs of some well-known enterprises

- Sun Microsystems: { iso(1) member-body(2) US(840) 113536 }
- RSA Data Security Inc: { iso(1) member-body(2) US(840) 113549 }
- Microsoft: { iso(1) member-body(2) US(840) 113556 }
- Netscape: { joint-iso-ccitt (2) country (16) usa (840) org (1) 113730 }
- Verisign: { joint-iso-ccitt (2) country (16) usa (840) org (1) 113733 }
- Intel: { joint-iso-ccitt (2) country (16) usa (840) org (1) 113741 }

ASN.1 Structured Types I

SEQUENCE

Zürcher Hochschule Winterthur

Use: Collection of a moderate number of variables that may be of different type and whose order is significant.

Type Definition:

```
UserAccount ::= SEQUENCE {
     username    VisibleString,
     password    VisibleString,
     accountNr   INTEGER
}
```

Value Assignment:

```
myAccount UserAccount ::= {
     username    "steffen",
     password    "jane51",
     accountNr    4711
}
```

ASN.1 Structured Types II
SEQUENCE OF

Zürcher Hochschule Winterthur

Use: Collection of a large number of variables of the same type and whose order is significant.

Type Definition:

```
MemberCountries ::= SEQUENCE OF VisibleString

AccountRegistry ::= SEQUENCE OF UserAccount
```

Value Assignment:

```
euMembers MemberCountries ::= {
              "Austria", "Belgium", "Denmark",
              "Finland", "France", "Germany",
              "Greece", "Ireland", "Italy",
              "Luxembourg", "The Netherlands",
              "Portugal", "Spain", "Sweden",
              "United Kingdom"}
```

ASN.1 Structured Types III

SET

Zürcher Hochschule Winterthur

Use: Collection of a moderate number of variables that may be of different type and whose order is insignificant.

Type Definition:

context-specific tags or automatic tagging

```
UserAccount ::= SET {
  username   [0] VisibleString,
  password   [1] VisibleString,
  accountNr  [2] INTEGER
}
```

Value Assignment:

```
myAccount UserAccount ::= {
  accountNr     4711,
  username      "steffen",
  password      "jane51"
}
```


Context-Specific Tags

- Since they can occur in arbitrary order, the members of a set need tags in order to uniquely identify them. Tags can either be defined explicitly by the system designer or are assigned automatically by the ASN.1 parser if the automatic tagging option is used.

ASN.1 Structured Types IV
SET OF

Zürcher Hochschule Winterthur

Use: Collection of variables that are the same type and whose order is insignificant.

Type Definition:

```
Keywords ::= SET OF VisibleString
```

Value Assignment:

```
someASN1Keywords Keywords ::=
                    {"INTEGER", "BOOLEAN", "REAL"}
```

Zürcher Hochschule Winterthur

Basic Encoding Rules of ASN.1
BER

General Encoding Rule for ASN.1 Values
Type - Length - Value

Zürcher Hochschule Winterthur

1. Primitive, definite-length encoding
   - simple types
2. Constructed, definite-length encoding
   - structured types (`SEQUENCE[OF]`, `SET[OF]`)
3. Constructed, indefinite-length encoding
   - structured types (`SEQUENCE[OF]`, `SET[OF]`)

ASN.1 Type Tags I
Universal Class Tags

Zürcher Hochschule Winterthur

UNIVERSAL 1	BOOLEAN	UNIVERSAL 2	INTEGER
UNIVERSAL 3	BIT STRING	UNIVERSAL 4	OCTET STRING
UNIVERSAL 9	REAL	UNIVERSAL 10	ENUMERATED

UNIVERSAL 6	OBJECT IDENTIFIER
UNIVERSAL 7	ObjectDescriptor
UNIVERSAL 26	VisibleString
UNIVERSAL 5	NULL
UNIVERSAL 23	UTCTime
UNIVERSAL 24	GeneralizedTime

...

UNIVERSAL 16	SEQUENCE [OF]	UNIVERSAL 17	SET [OF]

ASN.1 Type Tags II
Application Class Tags

Zürcher Hochschule Winterthur

Explicit Tagging: (strong type checking)

```
IpAddress ::= [APPLICATION 0]
                  OCTET STRING (SIZE(4))

Counter   ::= [APPLICATION 1]
                  INTEGER (0..4294967295)
```

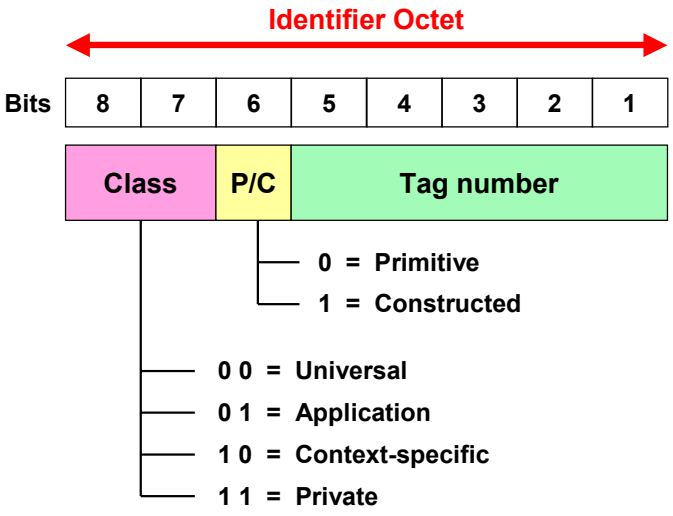
Implicit Tagging: (shorter encoding)

```
IpAddress ::= [APPLICATION 0] IMPLICIT -- RFC 1155
                  OCTET STRING (SIZE(4))

Counter   ::= [APPLICATION 1] IMPLICIT -- RFC 1155
                  INTEGER (0..4294967295)
```

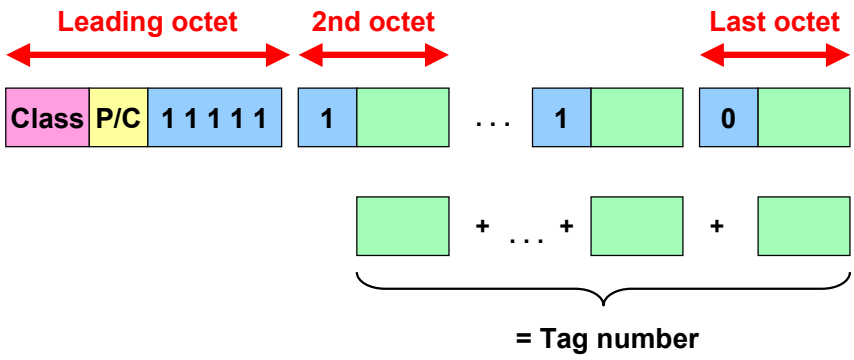
BER Encoding of Identifier Field I

Tag numbers < 31



BER Encoding of Identifier Field II

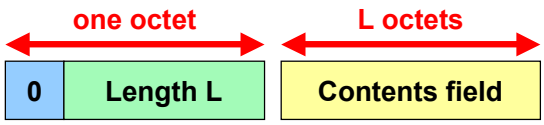
Tag numbers ≥ 31



BER Encoding of Length Field

Zürcher Hochschule Winterthur

- **Short definite form (L < 128 octets)**



- **Long definite form (128 ≤ L < 2^{1008} octets)**

- **Indefinite form; content field terminated by EOC**

Short definite form

- Used either for **primitive** or **constructed** types with content lengths smaller than 128 octets.

Long definite form

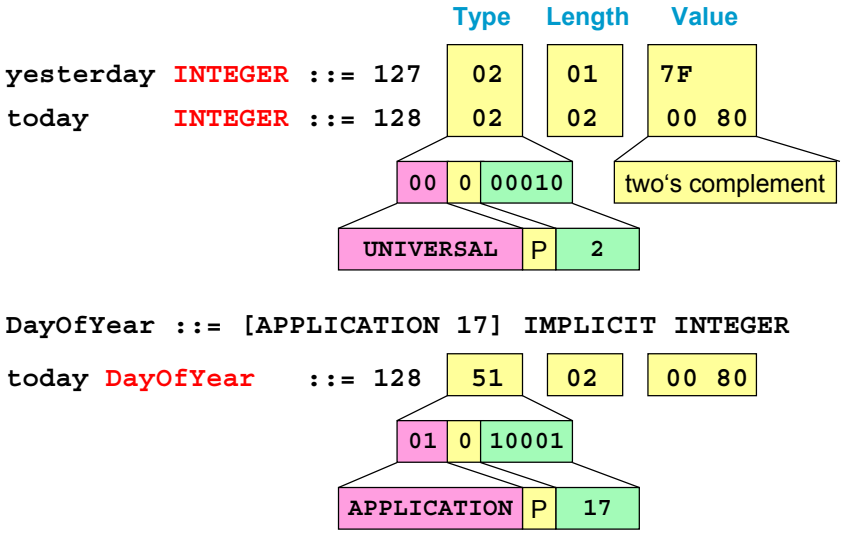
- Used either for **primitive** or **constructed** types with content lengths usually equal or greater than 128 octets.

Indefinite form

- Used for **constructed** types only.
- The end-of-contents octets (EOC) can be considered as the encoding of a value whose tag is universal class, whose form is primitive, whose tag number is zero, and whose contents are absent, i.e. it has zero length.

BER Encoding Examples I

INTEGER



BER coding of two's complement integers

- -129: 1111 1111 0111 1111 = 02 02 FF 7F
- -128: 1111 1111 1000 0000 = 02 01 80
- -127: 1111 1111 1000 0001 = 02 01 81
- -1: 1111 1111 1111 1111 = 02 01 FF
- 0: 0000 0000 0000 0000 = 02 00
- 1: 0000 0000 0000 0001 = 02 01 01
- 127: 0000 0000 0111 1111 = 02 01 7F
- 128: 0000 0000 1000 0000 = 02 02 00 80
- 129: 0000 0000 1000 0001 = 02 02 00 81

BER Encoding Examples II

SEQUENCE

Zürcher Hochschule Winterthur

Type Definition

```
Birthday       ::= SEQUENCE {
      name     VisibleString,
      day      DayOfYear
}
```

UNIVERSAL 16
00 1 10000

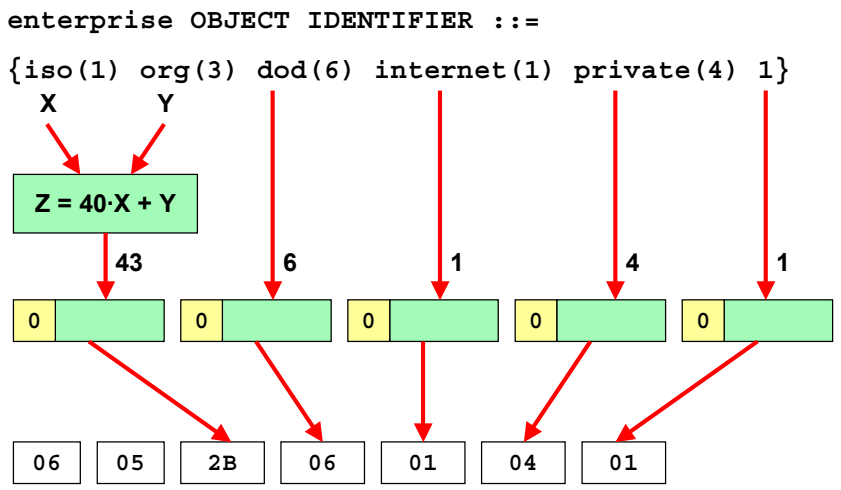
Value Assignment

```
myBirthday Birthday ::= {
      name     "Jane",
      day      128
}
```

BER Encoding

Birthday	Length	Contents		
30	0A			
		VisibleString	Length	Contents
		1A	04	"Jane"
		DayOfYear	Length	Contents
		51	02	00 80

BER Encoding Examples III
OBJECT IDENTIFIER



Coding of OID Root

- ccitt(0): Z = Y { 0 ..39}
- iso(1): Z = 40 + Y {40 ..79}
- joint-isi-ccitt(2): Z = 80 + Y {80 ..119}

Coding of OID node numbers

- Similar coding as tag numbers in identifier field
- Range {0..127}: `0xxx xxxx`
- Range {128..16383}: `1xxx xxxx 0xxx xxxx`
- Range {16384..2097151}: `1xxx xxxx 1xxx xxxx 0xxx xxxx`

Alternative Encoding Rules

Zürcher Hochschule Winterthur

- **Packed Encoding Rules (PER)**
  - Very compressed encoding based on ASN.1 subtype information. Example: subtype INTEGER (998..1001) is encoded using two bits only. Used e.g. in H.323 multimediastandard.
- **Distinguished Encoding Rules (DER)**
  - Subset of BER that gives exactly one way to represent any ASN.1 value. Used e.g. in X.509 certificates where computation of digital hash sums must be unique. Uses definite-length encoding.
- **Canonical Encoding Rules (CER)**
  - Subset of BER that gives exactly one way to represent any ASN.1 value but based on indefinite-length encoding.