Svibanj, 2004.

WHITE PAPER

ASN1
Abstract Syntax Notation One

Ivan Krsnik
0036386649

Sadržaj

1) Uvod
2) Povijest ASN1 notacije
3) Što je to ASN1
4) OSI model i ASN1
5) Prvi koraci sa ASN1
6) Osnove ASN1
   - Neka leksičko-sintaktička pravila
   - Tipovi podataka
   - Varijable
   - Informacijski objekti
   - Moduli i specifikacije
7) BER (Basic Encoding Rules)
8) PER (Packed Encoding Rules)

1) Uvod

Zamislite kompliciranе podatke neke velike svjetske banke, podatke neke aviokompanije, koja vodi brigu o rutama kojima avioni voze, o osoblju, stanju aviona, aerodromima, datoteke građevinske firme koja gradi nebodere od 400 m ili mostove od 30 km, raznorazne podatke velike automobilske firme npr. BMW... To su jako složene podatkovne strukture, koje izmjenjujemo raznim računalima (različitih proizvođača), različitim programima i aplikacijama.

Da bi osigurali bezbrižan prijenos takvih složenih podatkovnih struktura, na najsigurniji, najpouzdaniji i najučinkovitiji mogući način, koristi se ASN1.

Ovaj whitepaper opisuje neke osnovne karakteristike ASN1 notacije.

2) Povijest ASN1 notacije

Priča o ASN1 počinje u ljeto 1982. godine. Mnogi ljudi koji su tada radili na razvoju standarda u aplikacijskom sloju (OSI model) uočili su identičan problem: podatkovne strukture postale su previše složene da dozvole naručene procedure kodiranja i dekodiranja u bitove i bajtove. Također, kad su kompajleri preuzeli asemblere, opće mišljenje je bilo da koderi moraju automatski biti generirani iz specifikacije (da pismo postane ekvivalentno računalnom programu).

James White i Douglas Steedman su neovisno predložili osnove notacije I algoritma koji bi definirao format za kodiranje bitova za e-mail Message Handling Systems (MHS) protokole. Ta notacija i shema za kodiranje bili su sposobni kodirati složene podatkovne strukture.

James White tada je radio na X.400 (MHS) projektu za Xerox korporaciju. Poslije sastanka sa Douglasom Steedmanom, 1984. godine izdana je notacija X.409 (`Message Handing Systems: Presentation Transfer Syntax And Notation'). Imala je 32 strane i osnovna svrha je bila da definira " prezentacijsku transfer sintaksu koristeći protokole aplikacijskog sloja u MHS; u arhitekturi open systems interconnection (OSI), prezentacijska transfer sintaksa se koristi da predstavlja informacije izmjenjene između aplikacijskih entiteta".

Definira built-in tipove ANY, BIT STRING, BOOLEAN, CHOICE, INTEGER, NULL, OCTET STRING, SEQUENCE, SEQUENCE OF, SET i SET OF, kao i character string tipove IA5String, NumericString, PrintableString, T61String, VideotexString, i vremenske tipove GeneralizedTime i UTCTime. Prezentacija svakog tipa uslijedila je nakon notacije varijabli I njihovih kodiranja, koje su se u to vrijeme zvali `standard representation of type'. Svaki vid kodiranja (primitivno ili složeno, poznate ili nepoznate duljine) već je bilo definirano.

X.409 notacija je bila totalno neovisna o MHS sustavu, dijelom i zbog toga što su objekti tretirani e-mail protokolom bili suviše kompleksni. Kao rezultat, mnogi timovi koji su radili na standardizaciji OSI aplikacija, skužili su da bi im to (X.409) moglo biti korisno.

Najviše zahvaljujući Johnu Larmouthu (Salford University, UK), notacija X.409 gotovo odmah je usvojena u 'OSI svijetu'. On razlikuje apstraktnu notaciju od transfer sintakse u dva različita dokumenta i predlaže termin 'ASN.1' , gdje brojka 1 označava da bimnoge druge notacije nakon ove mogle biti standardizirane (to se nije nikad dogodilo). Kad je ASN.1 prvi put izdan, mnogi su ga nazvali 'engleska verzija X.409 notacije'.

Iako tehnički identična notaciji X.409, dva ISO dokumenta bila su potpuno izmjenjena zbog potrebe uvođenja Prezentacijskog sloja. Također, postalo je nužno razlikovati dio informacije u aplikacijskom sloju od njene prezentacije, pa su uvedene mnoge apstraktne transfer sintakse.

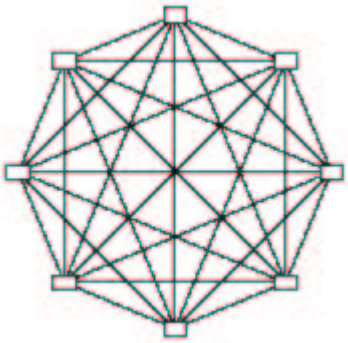
3) Što je to ASN1

Postoji puno različitih računalnih arhitektura, mnogo različitih programskih jezika koji imaju različite načine reprezentacije podataka.

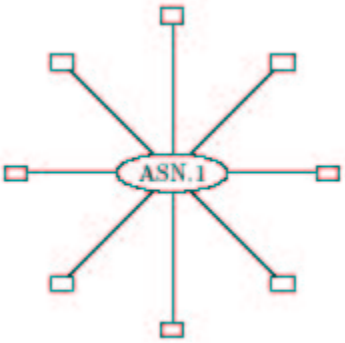
Računala mogu imati različitu internu reprezentaciju podataka koje pohranjuju. Tako npr. velika većina koristi ASCII kodiranje, dok neki koriste druge tipove kodiranja- EBCDIC kodiranje. Također, većina osobnih računala koristi 16 ili 32 bitnu riječ u dvokomplementarnoj aritmetici, ali postoje i oni koji koriste npr. 60 bitnu riječ i jednokomplementarnu aritmetiku.

Sve to govori u prilog tome da mora postojati neka univerzalna notacija prijenosa, te potreba za prevođenje u i iz te notacije. Ako želimo ostvariti međusobnu komunikaciju n računala koje imaju različite interne načine zapisivanja podataka, te ostvariti prijenos u oba smjera, trebamo imati n*(n-1) programa za konverziju (simetrična komunikacija.)

Postoje dva tipa komunikacije: smetrična i asimetrična.



Simetrična komunikacija



Asimetrična komunikacija

Pomoću telekomunikacijskih protokola prenosimo razne podatke, a bez obzira na to koji programski jezik koristimo, moramo **opisati** podatkovne strukture koje prenosimo. Taj opis bi morao poštivati sva moguća pravila određenog programskog jezika, ali bi morao ostati neovisan o programskm jeziku i nikad ne bi smio biti direktno implementiran na računalu.

Upravo zbog tih razloga, takav opis zovemo abstract syntax , a jezik Abstract Syntax Notation One ili ASN1.
Iako neovisan o programskom jeziku, ASN mora biti jednako moćan, mora omogućavati sastavljanje kompleksnih tipova podataka (analogija sa C-om→struct, list) od osnovnih (analogija sa C-om→integer, boolean,character string, bit string).

ASN1 je standard koji definira formalizam za specifikaciju apstraktnih tipova podataka. ASN mora biti formalan jezik, da se spriječe dvosmislenosti kod interpretacije i obrade sa računalnim alatima, a podaci se prenose na velikom broju aplikacija: internet, inteligentne mreže, mobiteli, VoiceOverIP, interaktivna televizija, te mnoge druge.
Sam ASN u osnovi nije abstract syntax, već jezik koji opisuje abstract syntax.

Mi od samo jednog ASN1 opisa podatka možemo taj podatak koristiti u bilo kojem programskom jeziku. *ASN kompajler* je računalni alat koji čita program zapisan u nekom računalnom jeziku (ASN1) i prevađa ga u jezik bliži samoj arhitekturi računala(C, C++, Java...)

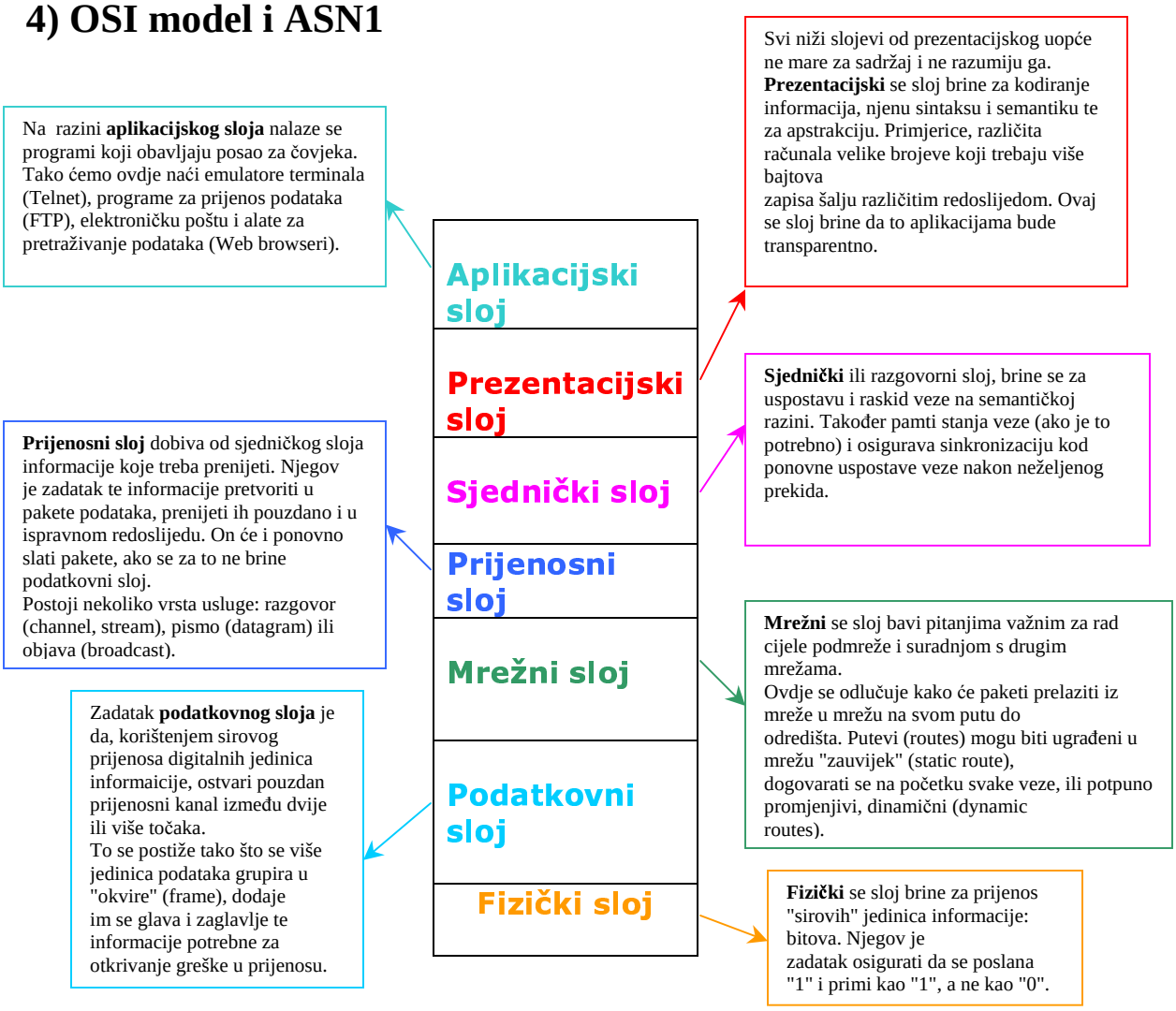
ASN1 šalje informaciju u bilo kojoj formi (audio, video, data) bilo gdje, gdje je digitalna komunikacija. ASN1 samo 'omata' neku informaciju koju prenosimo, pa zato to nije pravi programski jezik.

Jedan od glavnih razloga zašto je ASN1 uspio je taj što on podržava standardizirana pravila kodiranja BER (*Basic Encoding Rules*), PER (*Packed Encoding Rules*).Ta pravila opisuju kako se vrijednosti definirane u ASN1 kodiraju za prijenos, neovisno o računalu ili programskom jeziku.

ASN je mnogo učinkovitiji i ekonomičniji od konkurentskih notacija, omogućava pouzdan prijenos poruka, pa se koristi i u bežičnom prijenosu.

Alati na gotovo svim operacijskim sustavima podržavaju ASN1, koji je sveprisutan, a opet nevidljiv.

4) OSI model i ASN1



Open System Interconnect (**OSI**) je ISO standard za računalne komunikacije. To je **referentni model** prema kome bi trebalo projektirati i proizvoditi opremu i programe za računalne komunikacije. Ako je proizvod napravljen u skladu s ovom normom, onda se može koristiti u zajednici s proizvodima viših i nižih slojeva bilo kojeg proizvođaća.

Svaki sloj zapravo komunicira samo sa istim tim slojem na drugom računalu. Zadatke prima od višeg sloja, a komunikaciju obavlja izdajući zadatke nižem sloju.

Funkcije koje sustavi trebaju izvršiti kako bi se obavili poslovi putem računalne komunikacije podijeljeni su u ovom referentnom modelu u sedam slojeva: fizički, podatkovni, mrežni, prijenosni, sjednički, prezentacijski i aplikacijski.
Svaki sloj je neovisan o unutarnjim svojstvima, načinu realizacije, a time i proizvođaču svih ostalih slojeva. To vrijedi ne samo za isto računalo, već i za slojeve drugih računala. Čak i isti sloj na dva računala može imati različite proizvođače.
U osnovi, sloj komunicira samo sa slojem iste razine na korespondentnom računalu. Potpuno je nesvjesan postojanja slojeva drugih razina i svega što se poruci dešava na putu.
Razmjena podataka između prezentacijskog i aplikacijskog sloja upravljana je sa ASN1.

5) Prvi koraci sa ASN1

U prethodnom dijelu smo vidjeli glavne principe prijenosa podataka, posebno u kontekstu OSI modela. Sada ćemo u okvirima primjera iz 'realnog života' prikazati problem prijenosa podataka, te kako opisati te podatke u ASN1 notaciji.

Pretpostavimo da postoji neka firma koja distribuira sportsku opremu određenog proizvođača, te ima nekoliko dućana povezanih sa središnjim skladištem gdje se drži roba. U slučaju dobre prodaje određenog proizvoda iz nekog od dućana, proizvod se mora naručiti iz tog glavnog skladišta. Ta firma zahtjeva da se protokol naručbe robe sastoji od sljedećih koraka:

- Naručbe se skupljaju lokalno po dućanima
- Prosljeđuju se u glavno skladište, gdje se vrši organizacija isporuke
- Račun o isporuci se mora poslati u dućan

ASN1 oblik takvog protokola naručbe robe bi izgledao ovako:

```
Naručba DEFINITIONS AUTOMATIC TAGS ::=
BEGIN
--Komentar: sada ćemo se koncentrirati na naručbe uzete u dućanima. Naručba se može
podijeliti u dva dijela:
```

- glava→sadrži identifikacijske podatke dućana
- proizvod→proizvodi koje se naručuje.

To se u ASN1 ovako zapisuje:
(ASN1 ključne riječi su napisane velikim slovima)

```
Naručba ::= SEQUENCE {
                    glava      Naručba-glava,
                    proizvod   SEQUENCE OF Naručba-linija}
```

To se može shvatiti ovako: naručba je struktura(SEQUENCE) sa dvije komponente: prva se zove glava(počinje sa malim slovom), a odnosi se na tip podatka Naručba-glava (počinje sa velikim slovom), druga se zove proizvod, a odnosi se an listu (SEQUENCE OF) podataka koji su svi tipa Naručba-linija. Naručba-glava se sastoji od sljedećih elemenata:

```
Naručba-glava ::= SEQUENCE {
                     broj      Naručba-broj,
                    datum      Datum,
                    dućan      Dućan,
                     plaćanje  Plaćanje-metoda }

Naručba-broj::= NumericString (SIZE (12))

Datum ::= NumericString (SIZE (8)) -- MMDDGGGG
```

```
Dućan ::= SEQUENCE {
            ime     PrintableString (SIZE (1..20)),                  --OPTIONAL ----------
            ulica   PrintableString (SIZE (1..50)) OPTIONAL,   --znači da komponenta--
            poštanskibroj NumericString (SIZE (5)),              --ulica nije nužno--------
            grad     PrintableString (SIZE (1..30)),                 --prenesena----------
            država  PrintableString (SIZE (1..20))
            DEFAULT default-država }            --DEFAULT znači da ako komponenta--
default-država PrintableString ::= "Hrvatska"    --država nije prenesena,----------
                                                 --dobit će vrijednost----------
                                                 -- default-država----------


Način-plaćanja ::= CHOICE {
              ček            NumericString (SIZE (15)),
              kreditnakartica  KreditnaKartica,
              gotovina        NULL }

KreditnaKartica ::= SEQUENCE {
            tip            Tip-kartice,
            broj           NumericString (SIZE (20)),
            vrijediDoDatuma  NumericString (SIZE (6))} -- MMGGGG --

--postoji pet tipova kreditnih kartica koje podržavamo
Tip-kartice ::= ENUMERATED { cb(0), visa(1), eurocard(2),
            diners(3), american-express(4) }

Naruđba-lista ::= SEQUENCE {
          kod-proizvoda    Kod-proizvoda,
          labela       Labela,
          količina     Količina,
          cijena       Lipe }

Kod-proizvoda ::= NumericString (SIZE (7))

Labela ::= PrintableString (SIZE (1..30))

Količina ::= CHOICE {jedinica [0] INTEGER,        --kako su sve tri alternative-----
                     milimetri [1] INTEGER,       --izbora CHOICE integeri,------
                     miligrami [2] INTEGER}       --moramo imati uglate----------
                                                  --zagrade--------------------------
Lipe ::= INTEGER
```

```
--Našu specifikaciju prijenosa robe zaključujemo sa izvještajem isporuke kojeg šaljemo
dućanu nakon narudbe

Isporuka-izvještaj ::= SEQUENCE {
            Narudba-kod   Narudba-broj,
            isporuka     SEQUENCE OF Isporuka-linija }

Isporuka-linija ::= SEQUENCE { proizvod  Proizvod-kod,
                               količina Količina }

END
```

6) Osnove ASN1

6.1 Neka leksičko-sintaktička pravila

ASN1 ključne riječi se pišu isključivo velikim slovima, osim nekih tipova character stringa, (npr. PrintableString, UTF8String...).
Postoje tri vrste stringova:

1) Character string u navodnicima
   ”This is a string”
2) Binarni string u navodnicima, sa velikim B na kraju:
   '01101'B
3) Heksadecimalni stringova unavodnicima, sa velikim H na kraju
   '0123456789ABCDEF'H

Za brojeve sa decimalnom točkom se ne koristi tip float kako ga znamo iz drugih programskih jezika (‘realni brojevi’ su u ASN1 formalno definirani sa tri integera: mantisa, baza i eksponent).

Komentari počinju sa dvostrukim minusom: --ovo je komentar , a završavaju ili na kraju reda, ili sa prvim duplim minusom prije kraja reda. Komentari se dodaju kako bi se neformalno opisalo što se radi u određenom dijelu programa, a što se ne može opisati i razumjeti iz samog koda jezika.

6.2 Tipovi podataka

Tip podataka je neprazan skup varijabli, koje se mogu kodirati za prijenos. Iako se može uspoređivati sa formalizmom poznatim iz npr. Pascala, za definiranje složenih tipova podataka, ASN1 tipovi podataka morali bi biti specifični za prijenos i omogućiti odgovarajuće funkcionalnosti (npr. OPTIONAL opcija u SEQUENCE tipu).

Sva ASN1 pridruživanja upotrebljavaju simbol ’::=’ .

Primjer:

```
Oženjen ::= BOOLEAN
Godine  ::= INTEGER
Slika    ::= BIT STRING
Forma   ::= SEQUENCE { ime PrintableString,
                       godine Godine,
                       brak Brak,
                       brak-potvrda Slika OPTIONAL }
```

BOOLEAN	Logičke vrijednosti **istina** i **laž**
NULL	Uključuje samu varijablu NULL, koja se upotrebljava za izvještaj o isporuci ili neke alternative CHOICE tipa (posebno za rekuzivne tipove podataka)
INTEGER	Pozitivni ili negativni cijeli brojevi
REAL	Realni brojevi
ENUMERATED	State of a machine ,npr.
BIT STRING	Bitovni string
OCTET STRING	Bajtovni string
OBJECT IDENTIFIER, RELATIVE OID	Jasna identifikacija entiteta registriranog u worldwide stablu
EXTERNAL, EMBEDDED PDV	Prezentacijski (šesti sloj) kontekst preklapajućih tipova podataka
...String	Različiti tipovi character stringova
CHARACTER STRING	Dozvoljava razmatranje specifične abecede za character string
UTCTime, GeneralizedTime	Datumi

Tablica osnovnih tipova podataka

CHOICE	Izbor između tipova podataka
SEQUENCE	Uređena struktura varijabli različitih tipova podataka
SET	Neuređena struktura varijabli različitih tipova podataka
SEQUENCE OF	Uređena struktura varijabli istih tipova podataka
SET OF	Neuređena struktura varijabli istih tipova podataka

Konstruktivni tipovi podataka

6.3 Varijable

Kod definiranja novih tipova podataka, vrijednosti varijabli koje pripadaju tim tipovima mogu biti eksplicitno definirane. Pridruživanje vrijednosti varijablama također upotrebljava simbol '::='.
Imena varijabli su riječi koje počinju sa malim slovom, te moraju pripadat određenom tipu podatka (ime počinje sa velikim slovom). Primjeri:

```
brojevi Lutrija-broj ::= 45
šesterostruki Lutrija-izvlačenje ::= { 7, 12, 23, 31, 33, 41 }
par Koordinate ::= { x 5, y -3 }

gornja-granica INTEGER ::= 12
Interval ::= INTEGER (0.. gornja-granica)
default-vrijednost Interval ::= 0
Par ::= SEQUENCE {
                    prvi [0] Interval DEFAULT default- vrijednost,
                    drugi [1] Interval DEFAULT default- vrijednost }
```

6.4 Informacijski objekti

Ponekad se potrebno izražavati formalnije nego u komentarima, semantičke veze koje postoje između tipova podataka i varijabli definiranih u specifikacijama. Nekad želimo pokazati činjenicu da komponenta u strukturi ovisi o varijablama pridruženih toj strukturi. Semantičke veze su formalizirane sa klasama informacijskih objekata. Za takvo 'klasno' pridruživanje koristi se simbol '::=' iza čega slijedi ključna riječ CLASS. Ime te klase se mora pisati sa velikim slovima. Primjer:

```
OPERACIJA ::= CLASS {
&broj INTEGER UNIQUE,
&Argument-tip,
&Rezultat-tip }
WITH SYNTAX { OPERACIJA BROJ &broj
                    UZIMA ARGUMENT  TIPA &Argument-tip
                    I VRAĆA VRIJEDNOST TIPA &Rezultat-tip }
```

Ovakva klasa definira specifičnu operaciju sa jedinstvenim brojem, povezujući svaki od njih sa svojim tipom argumenta i tipom rezultata. Ovdje se ne govori o tome što određena operacija radi, jer to ne spada u ASN1, već specificira tipove podataka koji se izmjenjuju kad jedno računalo zatraži drugo računalo da izvodi takve operacije.
Polje &broj počinje sa malim slovom, a iza toga slijedi tip podatka→INTEGER, te se koristi kao identifikacija za informacijski objekt (zbog toga jer nakon integera dolazi ključna riječ UNIQUE).
Polja &Argument-tip i &Rezultat-tip počinju velikim slovima (ali nakon toga ne dolazi ništa); oni se odnose na bilo koji ASN1 tip podatka.
Blok WITH SYNTAX definira više user-friendly sintaksu da označi objekte ove klase.

6.5 Moduli i specifikacije

Pošto smo se upoznali sa raznim konceptima koji se mogu definirati sa ASN1 notacijom, sada ćemo vidjeti kako ih sakupiti da napravimo specifikaciju prijenosa podataka komunikacijskim protokolima. **Specifikacija** se sastoji od nekoliko ASN1 modula, a svaki **modul** skuplja tipove podataka (uglavnom), varijable, klase informacijskih objekata, informacijske objekte.

Ime modula počinje sa velikim slovom. Može biti referenciran kao neka vrsta 'univerzalnog pokazivača' , u vitičastim zagradama iza imena. Evo jednog primjera takvog modula:

```
Module2 { iso member-body(2) f(250) type-org(1) ft(16)
          asn1-book(9) chapter5(0) module2(1) }
DEFINITIONS AUTOMATIC TAGS ::=
BEGIN
EXPORTS Type2;
IMPORTS Type1, value FROM Module1 {iso member-body(2)
                                   f(250) type-org(1) ft(16) asn1-book(9)
                                   chapter5(0) module1(0)};
Type2 ::= SEQUENCE OF Choice
Choice ::= CHOICE { a INTEGER (0..value),
                    b Type1 }
END
```

Dio AUTOMATIC TAGS nam govori da me trebamo posebno označavati komponente strukture (sa npr. uglatim zagradama "[" i "]") koje su definirane kasnije u modulu, za to brine kompajler.

Dijelovi IMPORTS i EXPORTS definiraju sučelje modula.
Dio IMPORTS sadrži prijenosna imena definirana u ostalim modulima i uvodi ih u tekući modul.
Dio EXPORT sadrži prijenosna imena koja bi trebala biti dostupna izvan tekućeg modula i zbog toga mogu biti uvezena unutar nekog drugog modula.

Može se zaključiti da sva prijenosna imena koja se pojavljuju u ASN1 služe samo za povezivanje (u bilo kojem smislu) sa podacima koje prenosimo.

Neke karakteristike ASN1 su specifične za prijenos podataka→ ne pojavljuju niti u jednoj apstraktnoj notaciji ili programskom jeziku.

7) BER (Basic Encoding Rules)

BER sintaksa za prijenos uvijek ima format TDV < Tip, Dužina, Varijabla >. Sva polja T, D i V su bajtovi. I samo polje V može biti u TDV formatu.

T	D	V
Tip okteti	Dužina okteti	Varijabla okteti

Format TDV

Rekurzivni princip

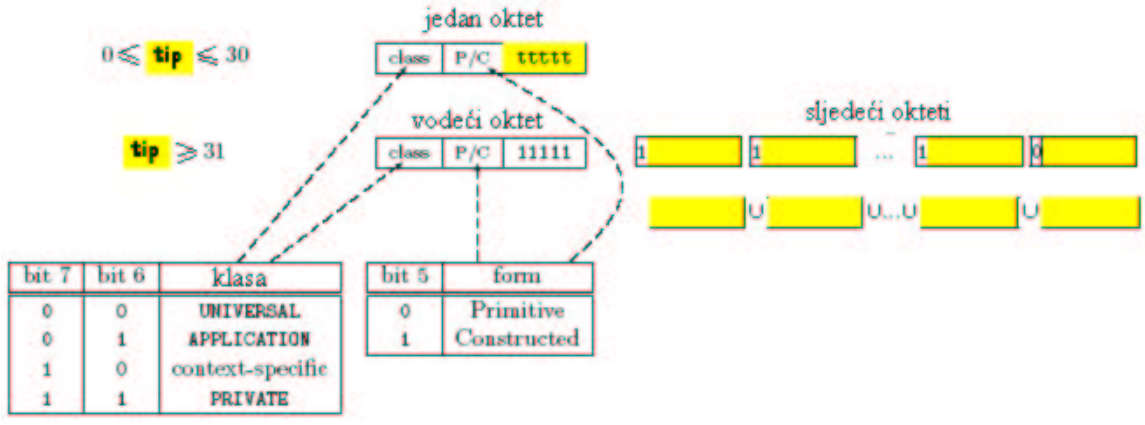
76543210
01011001

Težine bitova

BER sintaksa prijenosa (TDV format)

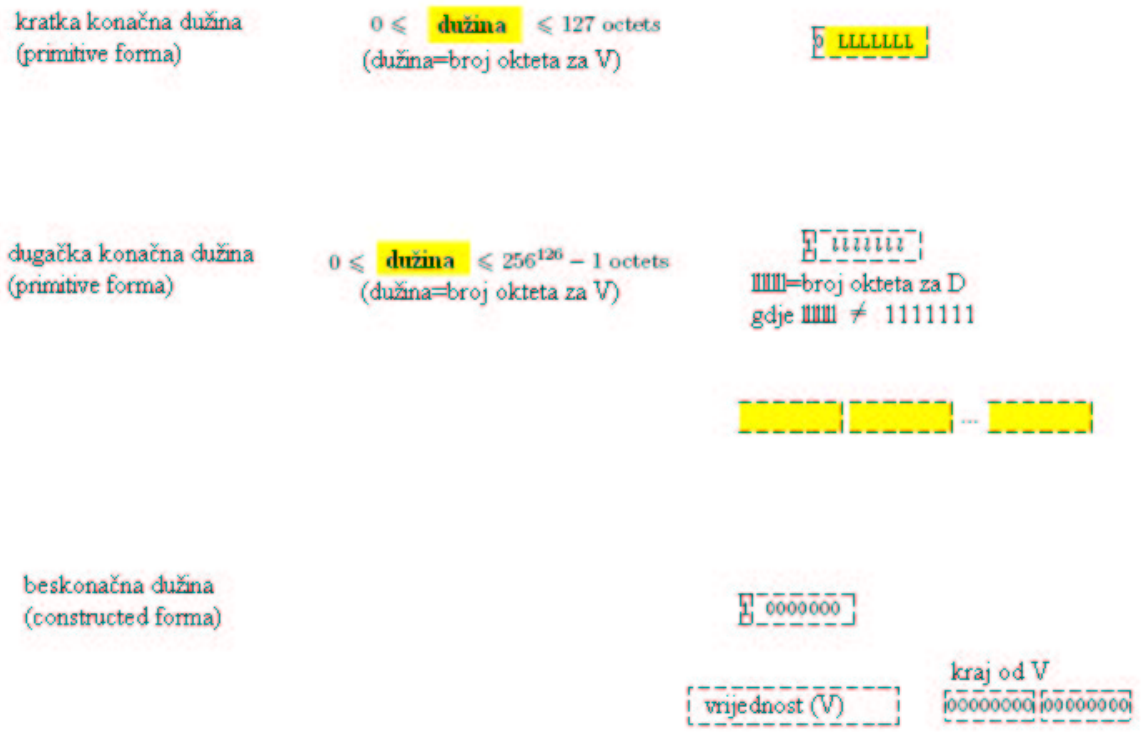
(T) Okteti Tip (generalno je dovoljan jedan oktet) se podudaraju sa kodiranjem oznake koja predstavlja kojeg je tipa varijabla (V). Jedan od bitova definira formu (basic ili constructed) V okteta.

Ako je broj tipa manji ili jednak 30, klasa i broj tipa su kodirani u jedan oktet, a ako broj tipa veći od 30, tada je drugačije, kao što se vidi na slici ispod:



Dva formata tipa (T)

(D) Okteti dužina predstavljaju dužinu V-a koji se kodira. Ako bit broj 5 u prvom T oktetu indicira primitive formu kodiranja, D je kodiran u konačnoj formi. Ako je forma kodiranja constructed, tada pošiljaoc bira da li se D kodira u konačnoj ili beskonačnoj formi.



Tri formata dužine (D)

U dugačkom formatu, prvi oktet dijela D predstavlja dužinu dužine (broj okteta potreban za kodiranje dužine; ne može biti kodirana u 127 okteta).

Kodiranje beskonačne forme se koristi kad se dužina V-a ne zna prije nego što je poslan cijeli TDV.

8) PER (Packed Encoding Rules)

Loše strane BER-a su troškovi u smislu veličine (50% veći od samog podatka kojeg kodiramo). To je dovelo do razvoja PER-a 1994. godine. Dobitak u veličini je najmanje 40 do 60% u usporedbi sa BER-om.

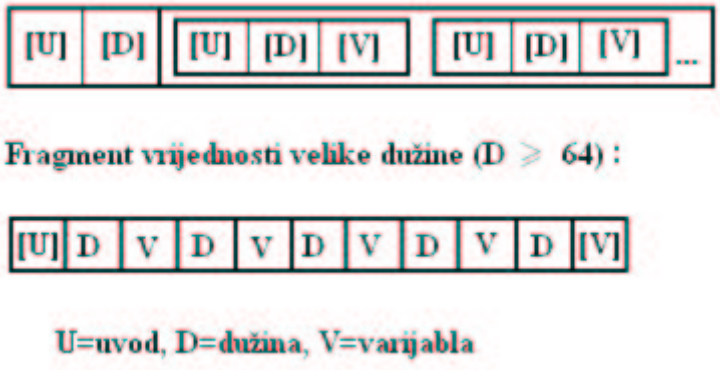
Zbog toga su ta pravila kodiranja pogodna kod protokola koji trebaju prenosit podatke velikom brzinom, posebno u domenama kao što je telefonija preko interneta, videofonija i multimedija općenito.

Zlatno pravilo PER-a se može izreći u ovoj rečenici: 'omogućiti što kompaktnije kodiranje uz što je moguće jednostavnija pravila kodiranja'. Da se to napravi, alat za generiranje PER kodera i dekodera je uže i preciznije vezan za ASN.1 nego BER.

Umjesto da upotrebljava format TDV kao u BER-u, PER format se može interpretirati kao '[U][D][V]' <opcionalni uvod, opcionalna dužina, opcionalna varijabla> , gdje polja U, D i V više nisu serije bajtova, nego su to sad serije bitova.

	BER	**PER**
INTEGER (123456789..123456792) ::= 123456790	6 bajtova	2 bita
INTEGER (123456789..MAX) ::= 123456790	6 bajtova	2 bajta
INTEGER ::= 123456790	6 bajtova	5 bajtova
IA5String (SIZE (4)^FROM ("ACGT")) ::= "TGAC"	6 bajtova	1 bajt
IA5String (FROM ("ACGT")) ::= "TGAC"	6 bajtova	2 bajta
IA5String (SIZE (4)) ::= "TGAC"	6 bajtova	4 bajta
IA5String ::= "TGAC"	6 bajtova	5 bajtova
SEQUENCE OF BOOLEAN ::= {-- 64 elementa --}	195 bajtova	9 bajtova
SEQUENCE SIZE (64) OF BOOLEAN ::= {-- 64 elementa --}	195 bajtova	8 bajtova
SEQUENCE OF INTEGER (0..65535) ::= {-- 64 elementa --}	195 bajtova	129 bajtova
SEQUENCE { a INTEGER (0..7), b BOOLEAN, c INTEGER (0..3), d SEQUENCE { d1 BOOLEAN, d2 BOOLEAN }} ::= { a 5, b TRUE, c 1, d { d1 TRUE, d2 TRUE }}	19 bajtova	1 bajt

Usporedba veličina kodiranja između BER-a i PER-a



Rekurzivni format PER prijenosne sintakse

Sada je jasno zašto se tipovi (iz BER-a) nikad ne kodiraju u PER-u. Polje dužine D se kodira jedino ako veličina nije navedena kao SIZE podtip (subtype constraint) u ASN.1 specifikaciji ili ako je veličina podatka bitna.

Kodiranje varijabli tipa SEQUENCE ili SET prethodi bitmapom koja pokazuje odsutnost ili prisutnost opcionalnih komponenti. Slično, indeks indicira da je alternativa sačuvana u tipu CHOICE prije kodiranja vrijednosti vezane sa tom alternativom.

Iako im to nije bila osnovna namjena, PER koderi i dekoderi troše manje procesorskog vremena nego BER koderi i dekoderi. Prijenos je brži nego BER jer najniži slojevi mrežekoriste kraće pakete.

Na tržištu ima malo PER kompajlera, vjerojatno zato jer ih je teže razviti.