P R O G R A M S K I
J E Z I C I

Uvod u XML
i
XML tehnologije

Dr Milica Vučković

Sadržaj

- Osnove XML-a
- Validacija XML dokumenata
- XSLT transformacije
- Modeli XML parsera
- MS implementacija W3C XML standarda

Ključni koncepti XML

XML eXtensible Markup Language

XML je danas postao *de-facto* standard za opis sadržaja i strukture (tekstualnih i multimedijalnih) dokumenata i razmenu dokumenata na Web-u

Ključni koncepti XML

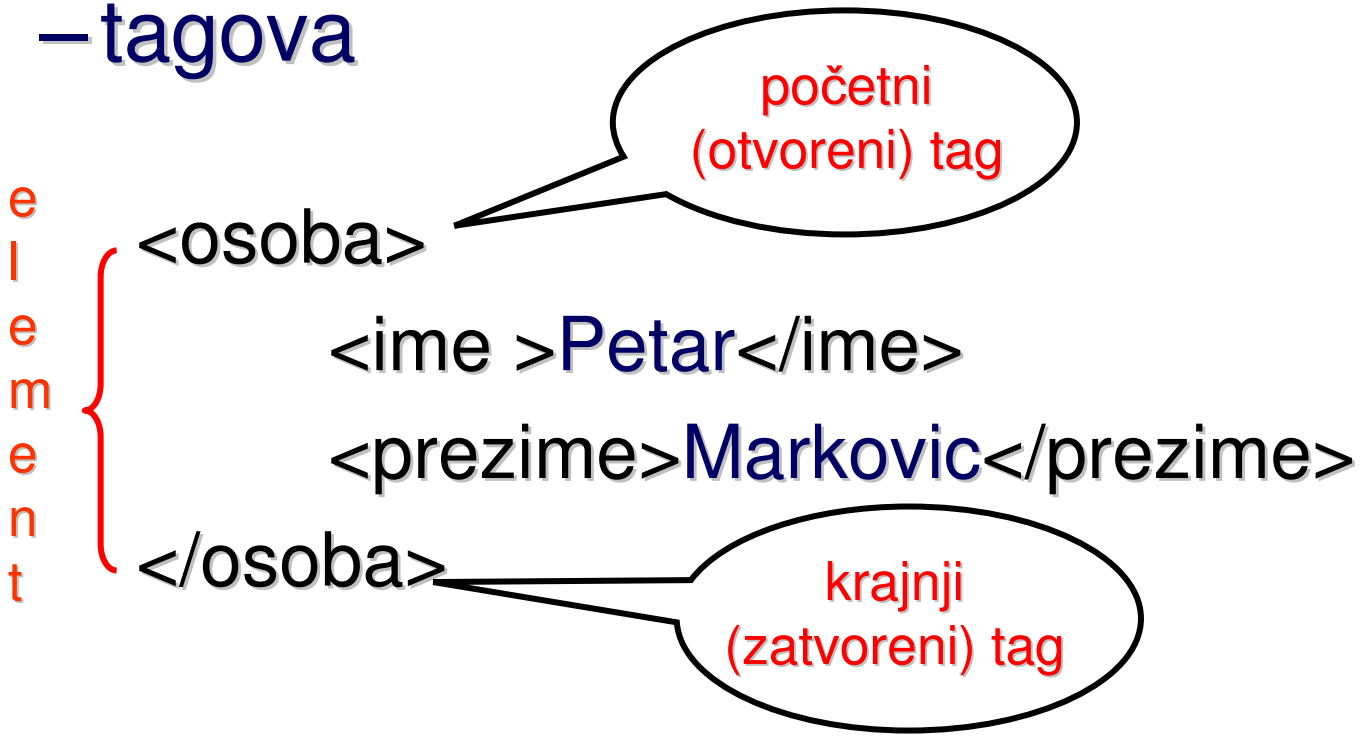
XML eXtensible Markup Language

- ***Markup***
  - dodavanje specijalnog značenja podatku
  - U XML koristi se *tag* za predstavljanje markup-a

- ***Extensible***
  - *proširljiv* jezik, dozvoljava definisanje novih tag-ova
  - meta jezik omogućava definisanje drugih markup jezika

Ključni koncepti XML

- XML dokument se sastoji iz
  - teksta
  - tagova



Ključni koncepti XML-a

- XML dokument: samoopisujuća, platformski nezavisna tekstualna datoteka
- Razdvajanje struktuiranog sadržaja dokumenta od njegove prezentacije (Style Sheet)

Korišćenje XML-a

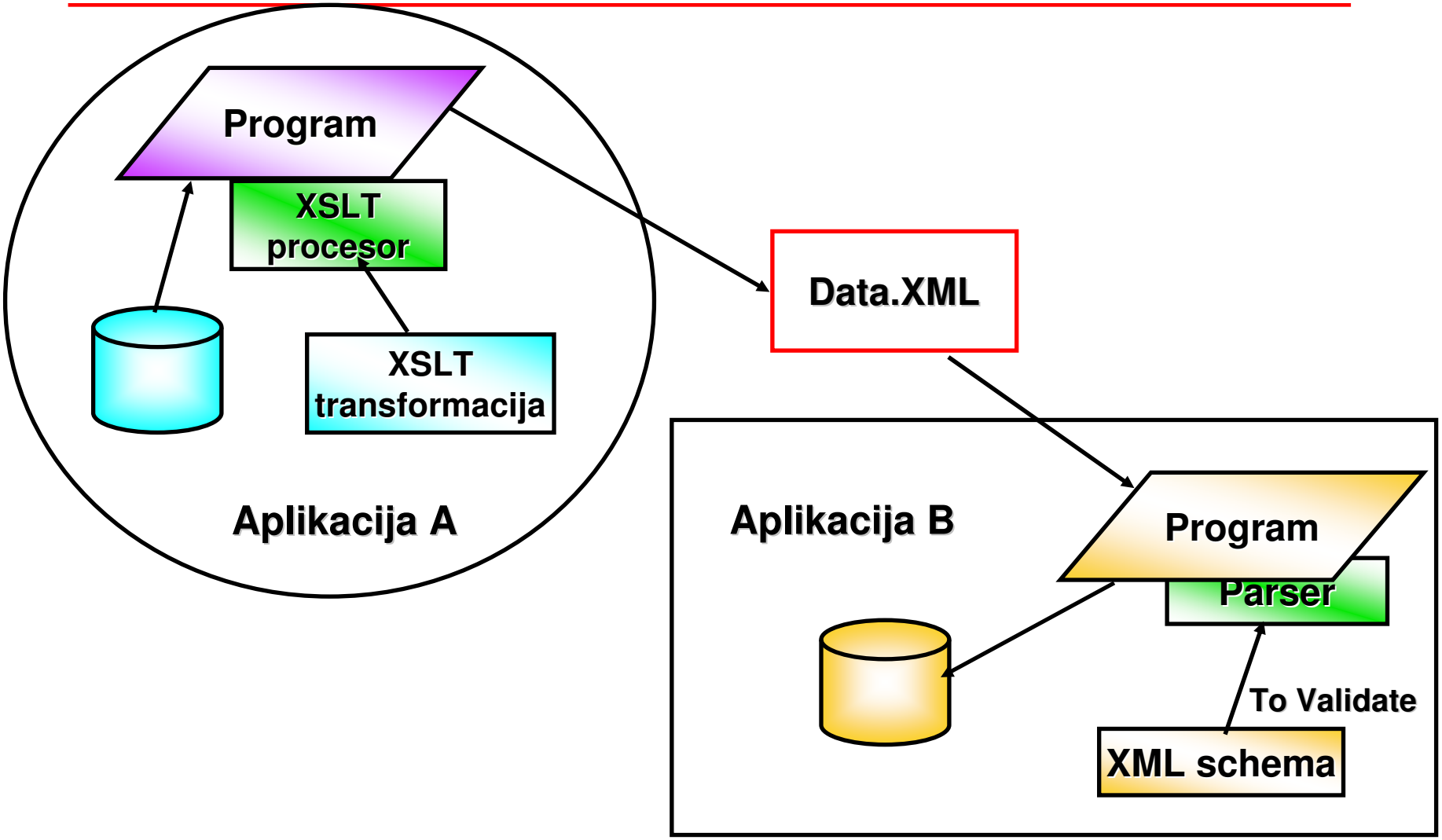
XML tehnologije:

- Procesiranje XML dokumenata
  - XML parseri
  - transfromacija XML dokumenata (XSLT jezik)
- Specifikacija logičke strukture XML dokumenata
  - DTD = Document Type Definiton
  - XML schema
- Upitni XML jezici
  - XPath
  - XQuery

Korišćenje XML-a

- XML je projektovan za distribuirano okruženje
  - XML je veoma pogodan kao format za razmenu podataka između heterogenih aplikacija na Web-u
  - XML kao format je dovoljno formalan za mašinsko procesiranje i dovoljno razumljiv za korisnike

Korišćenje XML-a



Korišćenje XML-a

- Web servisi

  nov standard za kreiranje interoperabilnih distribuiranih aplikacija

  Skup standarda zasnovani na XML-u
  - SOAP = Simple Object Access Protocol
  - WSDL = Web Service Definition Language
  - UDDI = Universal Description, Discovery and Integration Protocol

Korišćenje XML-a

- Memorisanje XML podataka:
  - XML datoteke i XML baze
  - realcione baze
    i. Transformacija XML dokumenta u relacione tabele
    ii. XML dokumenta memorišu se u kolone (čiji je tip -XML type) relacionih tabela

Istrorijski razvoj XML-a

W3C = World Wide Web Consortium

(organizacija za standardizaciju Web tehnologija)

- 1996. počela razvoj standarda za XML sa motivacijom da XML treba da kombinuje
  - Fleksibilnost **SGML**

    SGML **=** Standard Generalized Markup Language
  - jednostavnost **HTML**

    HTML = Hypertext Markup Language

- U februaru 1998. definisan je **XML 1.0 standard**

SGML

- standard za definisanje i reprezentovanje strukture različitih tipova elektronskih dokumenta, (ISO standard 1985)
  - tagovi se koriste samo za označavanje strukture dokumenta
  - proširljiv jezik, dozvoljava definisanje novih tagova
  - meta jezik standard za definisanje novih markup jezika
  - Veoma složen jezik

- XML je podskup SGML (prilagodjen potrebama Web-a)

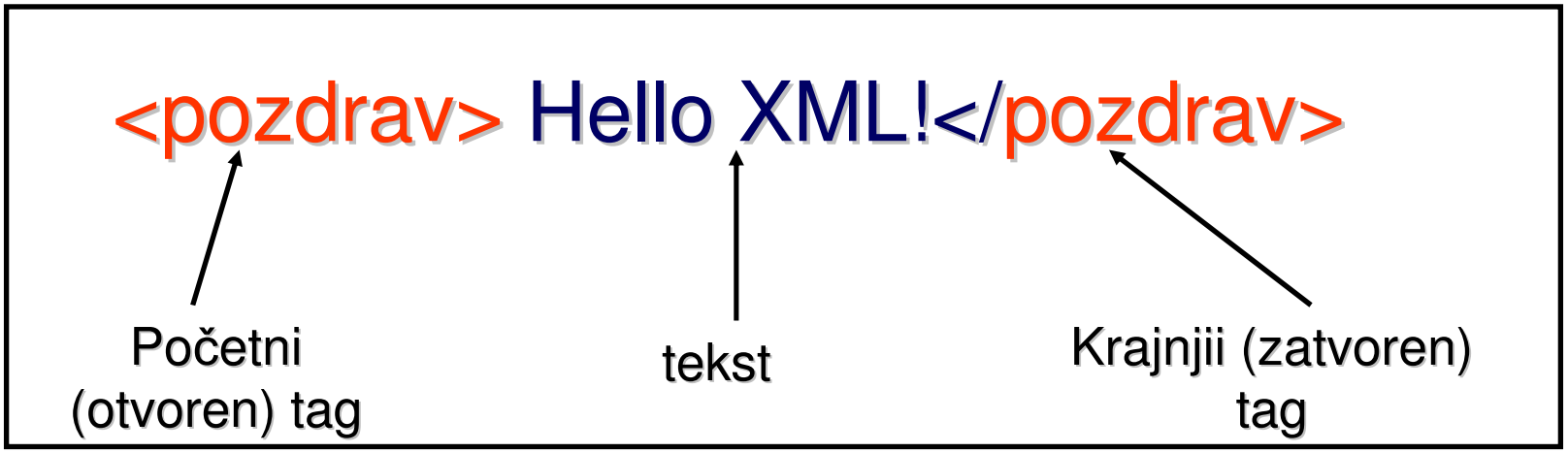
HTML

- Početkom 1990. HTML je definisan od W3C kao standard
  i. definisan u SGML
  ii. koristi *fiksni* broj *predefinisanih* tagova
  iii. prvenstveno projektovan za formatiranje i prezentovanje dokumenta na Web-u

Primeri	*Prikaz u browser-u*
bold	**bold**
italic	*italic*

XML dokumenta

- XML dokumenta su samoopisujuće, platformski nezavisne tekstualne datoteke
- XML dokument sadrži :
  - Tekst (sadržaj dokumenta)
  - tag-ove



XML elementi

- Elementi su osnovni blokovi XML-a

Kontejner element par tag-ova (početni i krajnji tag) sa sadržajem

<pozdrav> Hello XML! </pozdrav>

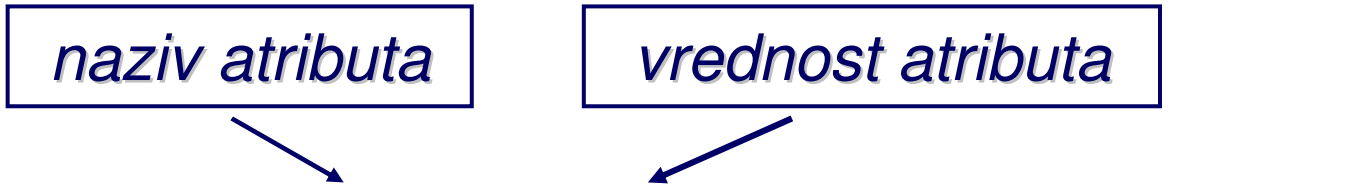
Prazan element obično se za krajnji tag koristi skraćenica />

<poruka/>

<pozdrav tekst = “Hello XML” />

XML atributi

- Elementima se mogu pridružiti atributi
  - obezbeđuju dodatne informacije o elementima




```
<poruka  datum = "12.5.08." sala = "201" >
  <tekst>Sastanak Katedre sutra u 10</tekst>
</poruka>
```

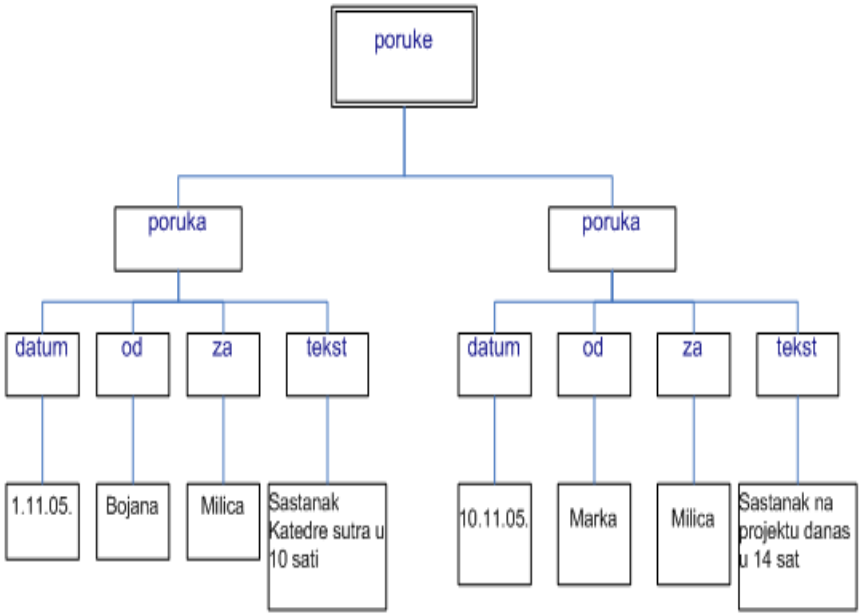
- Imena XML tagovi i imena atributa – *case sensitive*

Struktura XML dokumenta

- *hijerarhijska struktura* (stablo) koja se sastoji iz elemenata, atributa i znakovnih podataka
- XML dokument ima *jedan i samo jedan koreni (root) element*
- Svi ostali elementi u strukturi su elementi "deca" korenog element
  - dozvoljeno višestruko ugnježdavanje elemenata

Hijerarhijska struktura XML dokumenta

```
<?xml version="1.0" encoding="UTF-8"?>
<poruke>
   <poruka>
      <datum>1.11.05. </datum>
          <od>Bojana</od>
          <za>Milicu</za>
          <tekst>Sastanak Katedre sutra u 10 sati </tekst>
    </poruka>
   <poruka>
      <datum>10.11.05. </datum>
          <od>Marka</od>
          <za>Milicu</za>
          <tekst>Sastanak na projektu danas u 14 sati  </tekst>
    </poruka>
</poruke>
```



XML deklaracija

Svaki XML dokument mora da sadrži XML deklaraciju, tj. ***instrukciju obrade*** kojom se dokument identifikuje kao XML dokument.

- Osnovni oblik XML deklaracije:

```
<?xml version =“1.0”?>
```

- Opcioni oblik XML deklaracije:

```
<?xml version =“1.0” encoding= “UTF-8”?>
```

XML deklaracija

<?xml version =“1.0” encoding= “UTF-8”?>

- **?** oznaka za instrukciju obrade

  instrukcija obrade je poruka programima koji procesiraju XML dokument

- atribut **version** specificira XML verziju
- atribut **encoding** definiše znakovni kod u kome je XML dokument napisan

  UTF-8 (kompresovana verzija Unicode-a)

  UTF-16 (Unicode)

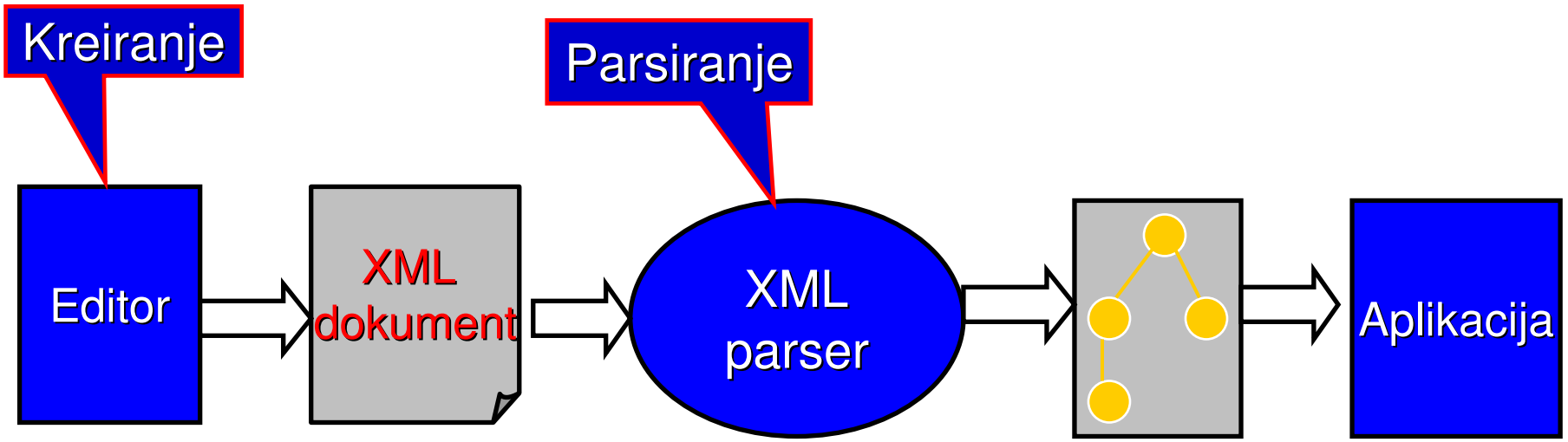
Dobro-oformljen XML dokument

- postoji XML deklaracija
- dokument sadrži jedan i samo jedan koreni element u kome su ugnježdeni svi ostali elementi i njihovi sadržaji
- svi elementi i atributi u dokumentu moraju da budu sintaksno ispravni

Provera sintaksne korektnosti XML dokumenta

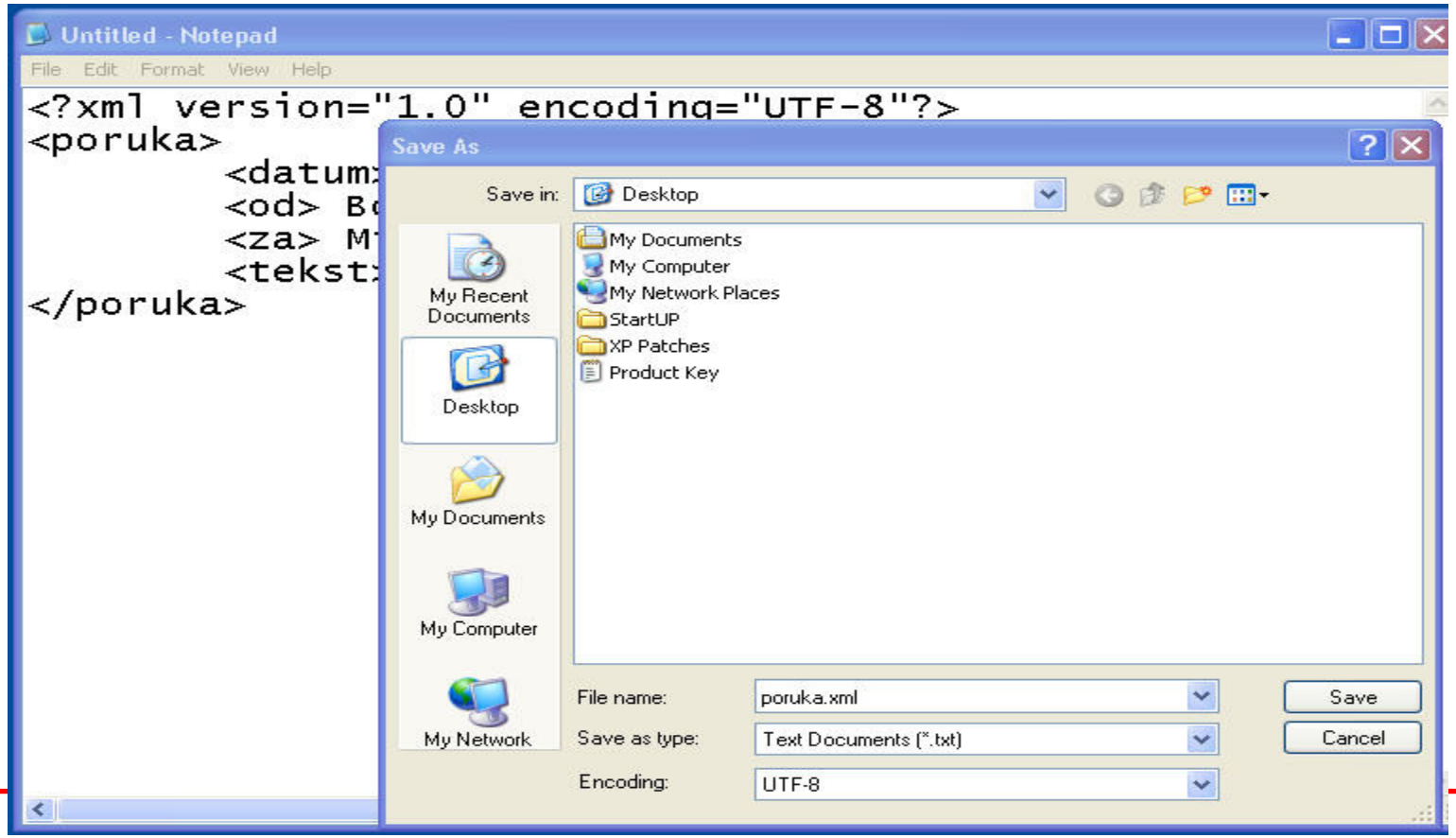
- ***XML parser*** verifikuje da li je XML dokument dobro-oformljen
- XML parser čita dokument i konvertuje ga u hijerahijsku strukturu
- XML parser prenosi parsirani dokument do krajnje aplikacije

Obrada XML dokumenta



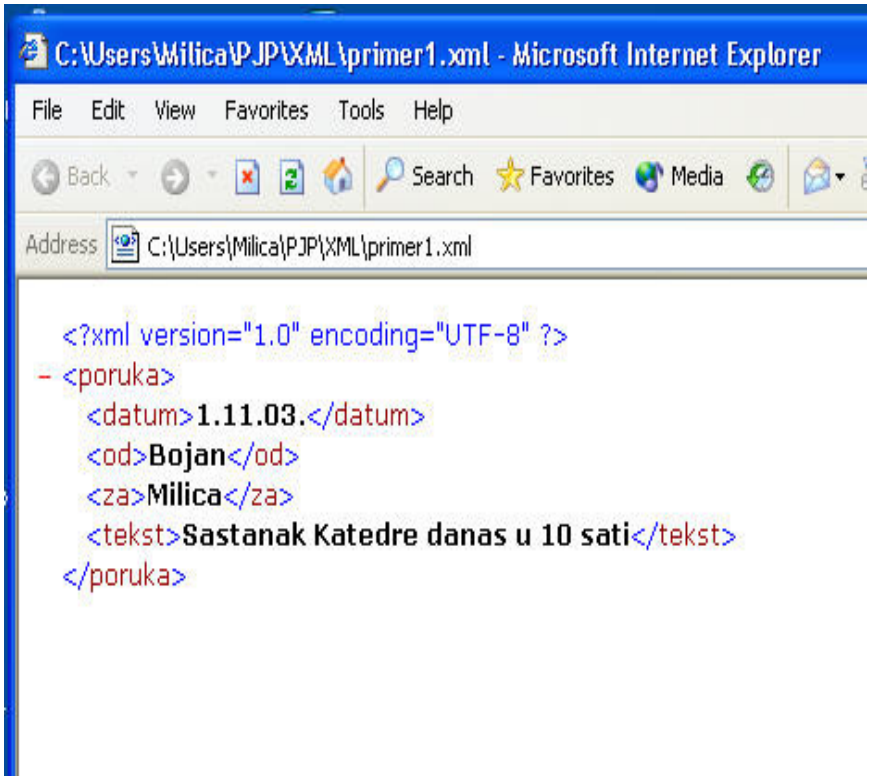
Kreiranje XML dokumenta

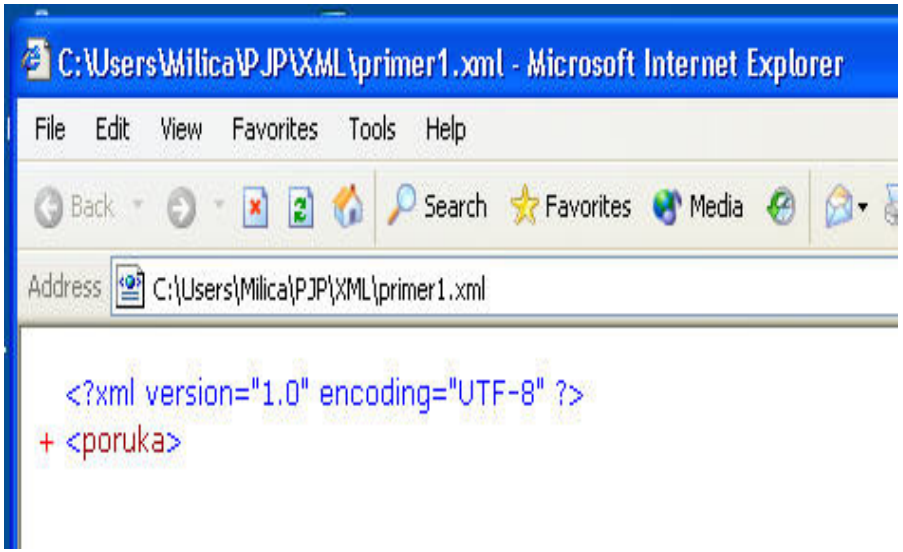
- Tekst editori (na primer Notepad)
- VS.NET XML Desinger
- XML Spy – razvojno okruženje za XML



Pregled XML dokumenta (source)

Pomoću web browser-a koji podržavaju XML
(Internet Explorer 5.0 i više verzije)





Validacija XML dokumenata

Definisanje tipova XML dokumenata

XML Schema

Validni XML dokument

- Dobro-oformljen
- Konzistentan sa strukturom definisanom u opisu tipa dokumenta

Definisanje tipova XML dokumenata

W3C je ponudio dva standarda načina za *definisanje* tipova XML dokumenta, odnosno *opisivanje* strukture XML dokumenta:

- *Document Type Definiton (DTD)*
- *XML Schema Definition (XSD)*

Definisanje tipova dokumenata

DTD i XSD definišu:

- strukturu XML dokumenta
- ime i tip svakog XML elementa/atributa

(DTD- ograničene mogućnosti za definisanje tipova)

DTD

- Nasledjen od SGML-a
- Poseban jezik
- Vrlo ograničene mogućnosti za definisanje tipova

Primer DTD

```
<!ELEMENT Knjige (Knjiga+)>
<!ELEMENT Knjiga (Naslov, Autor, Godina, ISBN, Izdavac)>
<!ELEMENT Naslov (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ELEMENT Godina (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Izdavac (#PCDATA)>
```

#PCDATA – Parser Character Data, oznacava znakovni sadržaj

“+” – element se pojavljuje bar jednom

XML Schema

- preporuka W3C od maja 2001
- data je preko XML sintakse (XML šema je XML dokument)
- podržava definicije prostih i složenih tipova i poseduje napredne mehanizme za grupisanje XML elemenata u XML dokumentu

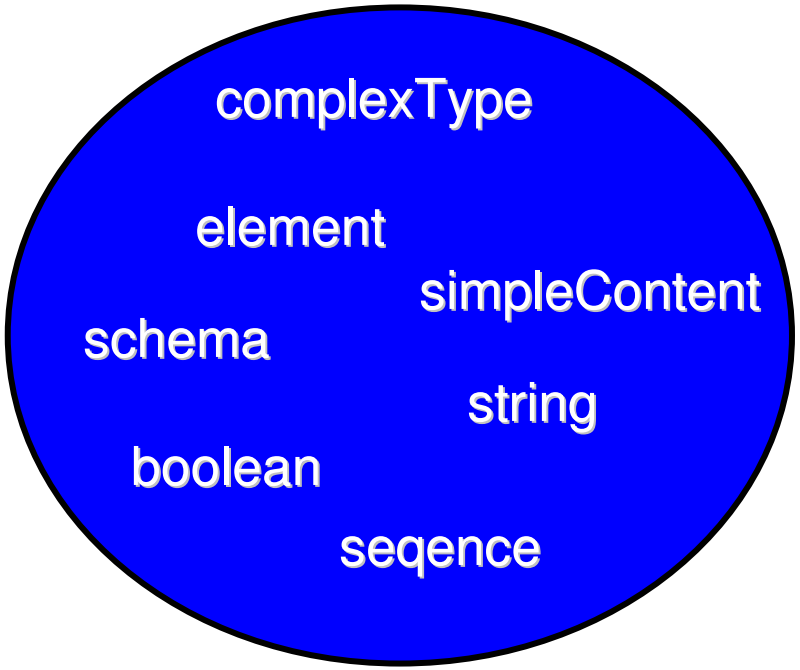
XML Schema

Za konstrukciju svake XML šeme koriste se:

- **Schema** element (koreni element svake XML šeme)
- Deklaracije elemenata
- Deklaracije atributa
- Definicije prostih i složenih tipova

Schema element

- *Schema* element ukazuje na *definiciju* XML šeme u kojoj se nalaze svi potrebni elementi za kreiranje XML šeme



Schema element

- Deklaracija XML namespace:

  xmlns:prefix=“namespace name”

  - prefix se koristi kao skraćeno ime za “namespace name” u XML šemi
  - “namespace name” je lokacija XSD i specificira se preko URL
- *Primer*

```
<?xml version="1.0"?>

<xsd:schema  xmlns:xsd=http://www.w3.org/2001/XMLSchema>

                    ……….

</xsd:schema>
```

Deklaracija elementa u XML šemi

- Za svaki *element* u XML šemi definiše se naziv i tip (atributi *name* i *type*)

  <xsd:element name =“Autor” type = “xsd:string”/>

  - Tip može da bude:
    - korisnički definisan tip (pr. ComplexType)
    - ili je u opsegu imena XML šeme (primer string)

  - Kardinalnost elementa može se specificirati u njegovom ocu-elementu; inače po difoltu, kardinalnost elementa je:

    minOccurs = “1”

    maxOccurs = “1”

Definisanje složenih tipova

Složeni tipovi se konstruišu od prostih i drugih složenih tipova korišćenjem konstruktora:

- **sequence** – def. uređenu grupu elemenata.

  Po difoltu, svaki element je obavezan (minOccurs = “1”) i jednoznačan (minOccurs = “1”)

- **choice** – def. grupu iz kojih se mogu izvlačiti pojedinačni elementi

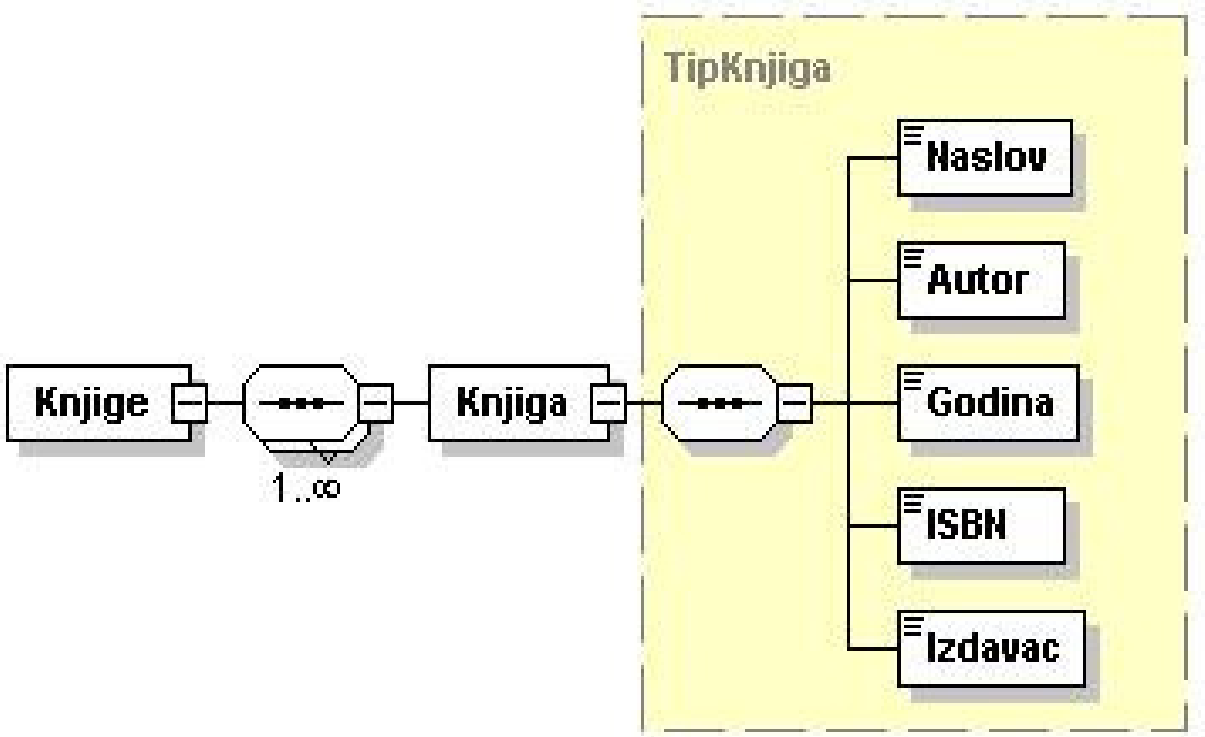
- **all** – def. grupu u kojoj se svi elementi mogu pojaviti maksimalno jedanput.

Primer XML šeme za tip Knjiga

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
   <xsd:element name="Knjige">
      <xsd:complexType>
         <xsd:sequence maxOccurs="unbounded">
            <xsd:element name="Knjiga" type="TipKnjiga"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="TipKnjiga">
      <xsd:sequence>
         <xsd:element name="Naslov" type="xsd:string"/>
         <xsd:element name="Autor" type="xsd:string"/>
         <xsd:element name="Godina" type="xsd:string"/>
         <xsd:element name="ISBN" type="xsd:string"/>
         <xsd:element name="Izdavac" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

knjige.xsd

Prikaz XML šeme preko strukture stabla



XML dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<Knjige xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Milica\FolderXML\knjige.xsd">
   <Knjiga>
      <Naslov>Baze podataka</Naslov>
      <Autor>Branislav Lazarevic</Autor>
      <Godina>2003</Godina>
      <ISBN>86-80239-96-8</ISBN>
      <Izdavac>FON, Beograd</Izdavac>
   </Knjiga>
   <Knjiga>
      <Naslov>Principles of Database and Knowledge-Base Systems</Naslov>
      <Autor>Jeffrey Ulman</Autor>
      <Godina>1998</Godina>
      <ISBN>0-7167-8158-1</ISBN>
      <Izdavac>Freeman</Izdavac>
   </Knjiga>
</Knjige>
```

XML dokument formiran u skladu sa XML šemom knjge.xsd

XSLT transformacije

- XSLT transformacioni proces
- XSLT stylesheet dokument
- XPath
- XSLT procesori

XSLT

XSLT eXtensible Stylesheet Language Transformation

- W3C standard

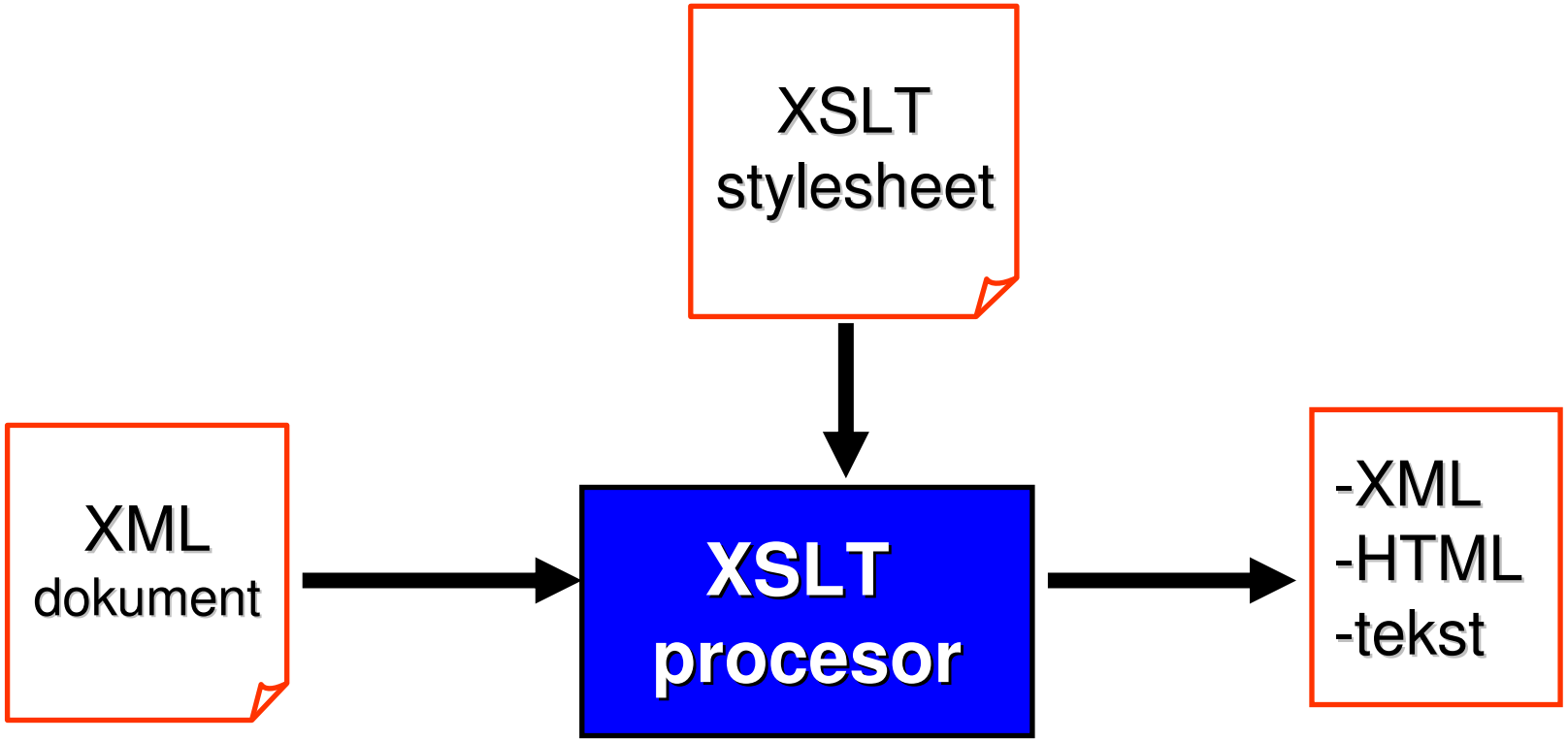
- XSLT verzije:
  - XSLT 1.0 (Novembar 1999)
  - XSLT 2.0 (Novembar 2002)

XSLT

XSLT je deklarativni jezik koji se koristi za opis pravila transformacija XML dokumenta u

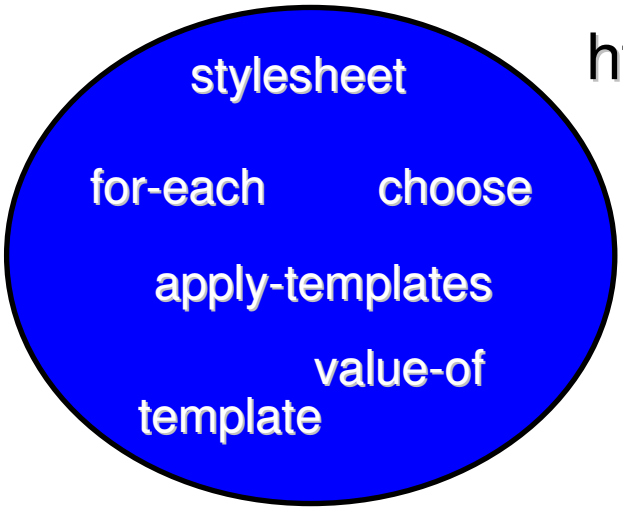
- drugi XML dokument
- HTML dokument
- tekst

XSLT transformacioni proces



XSLT stylesheet dokument

- XSLT stylesheet dokument je XML dokument
  - XSLT instrukcije se izražavaju kao XML elementi
- Elementi koji se koriste za konstruisanje stylesheet dokumenta definisani su preko XSLT namespace-a, lokacija se spec. preko URL-a:



http://www.w3.org/1999/XSL/Transform

Osnovne karakteristike XSLT stylesheet dokumenta

- **stylesheet** element je koreni element

```
<?xml version="1.0"?>
<xsl:stylesheet version "1.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
      .....
      .....
</xsl:stylesheet>
```

Osnovne karakteristike XSLT stylesheet dokumenta

- Stylesheet element sadrži skup templejt pravila koja se deklarišu sa **<xsl:template>** elementima

- pravila opisuju kako se pojedini elementi u XML dokumentu transformišu u rezultujuće elemente u izlaznom dokumentu

Templejt pravila

Templejt pravilo sadrži dva dela:

- ***pattern*** - identifikuje i izdvaja elemente ulaznog XML dokumenta na koje će biti primenjena transformacija
- ***akcija*** - opisuje transformaciju koja se primenjuje

```
<xsl:template match="pattern">
    [ akcija ]
</xsl:template>
```

Templejt pravila

<xsl:template match="*pattern*">

[akcija]

</xsl:template>

- match atribit koristi se za povezivanje templejta sa nekim delom ulaznog XML dokumenta
- Vrednost match atributa je XPath pattern

XPath

- **XPath** (***XML Path Language***) je jezik koji omogućava navigaciju do delova(čvorova) XML dokumenta (kao što su elementi, atributi, njihove vrednosti)

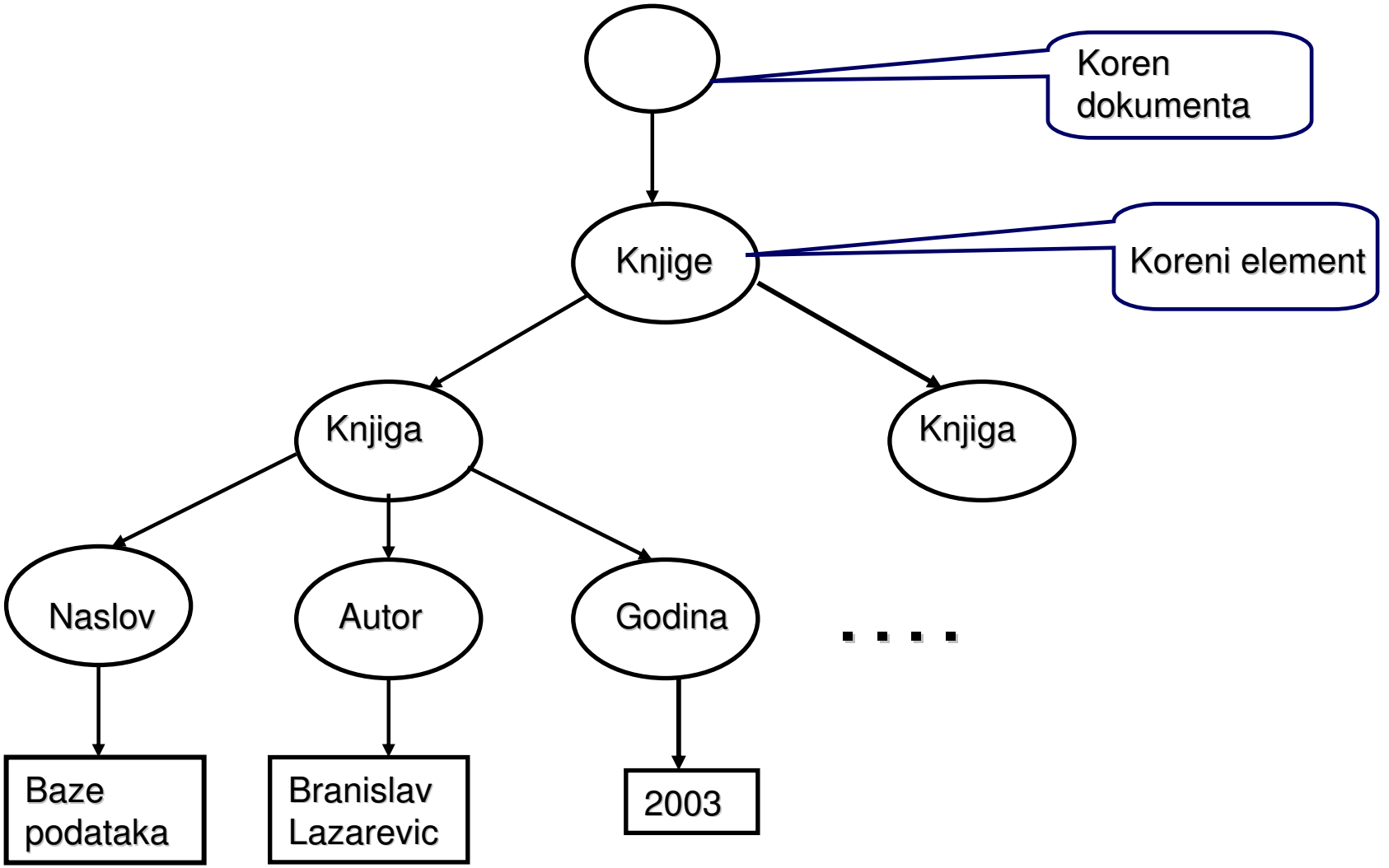
- XPath je W3C standard

- U XSLT dokumentu XPath se koristi za izdvajanje delova XML dokumenta na koje će biti primenjena transformacija

XPath

- XML dokument se ovde posmatra kao stablo čvorova sa definisanim čvorom koji se naziva ***koren dokumenta***
  - koren dokumenta je bezimeni čvor čije je dete koreni element XML dokumenta

Model podataka za XPath



XPath

- Čvor se adresira preko tzv. *izraza putanje* – niz od jednog ili više koraka razdvojenih sa "/"

- Primer

  /Knjige/Knjiga/Godina

  Rezultat: <Godina>2003</Godina>
  <Godina>1998</Godina>

  Izraz putanje koji pocinje sa "/" reprezentuje apsolutnu putanju

XPath

"/" XPath pattern za koren dokumenta

```
<xsl:template match="/">
  ..........
</xsl:template>
```

atribut match="/" povezuje templejt sa korenom dokumenta

Telo templejt pravila

Satoji se iz:

- **XSLT instrukcija**
- **Elemenata** koji specificiraju neki željeni izlazni tekst koji XSLT procesor treba da ubaci u izlazni dokument

Primer XSLT instrukcije value-of

```
<xsl:template match="/">
    <xsl:value-of  select = “pozdrav"/>
</xsl:template>
```

sadržaj elementa *pozdrav* prvo se dodeljuje atributu *select*
zatim, sadržaj elementa p*ozdrav* XSLT procesor kopira u izlazni dokument

Primer elemenata za specifikaciju izlaznog teksta

Pretpostavka: izlazni dokument je HTML dokument

```
<p> <font>
        <xsl:attribute  name="color">blue</xsl:attribute>
        <xsl:attribute  name="size">6</xsl:attribute>
                    pozdrav xml programera
        </font>
</p>
```

XSLT procesori

- Standalone XSLT procesori

  Java XSLT procesor, SAXON, Oracle XSLT, Xalan (Apache projekat)

- Korišćenje Web Browser-a za XSLT transformacije
  - MS Internet Explorer 5.5 i više verzije
    - XSLT procesor u IE je deo MSXML parsera
  - Netscape 6.0
  - JavaScript

- Korišćenje Web servera za XSLT transformacije
  - Tri načina za izvršavanje XSLT transformacija
    - Java servleti
    - ASP (Active Server Pages)
    - JSP (Java Server Pages)

Primer transformacije XML dokumenta u HTML dokument

XML dokument (pozdrav.xml):

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl“
        href=“pozdrav.xsl"?>
<Pozdrav>
        Hello XML!
</Pozdrav>
```

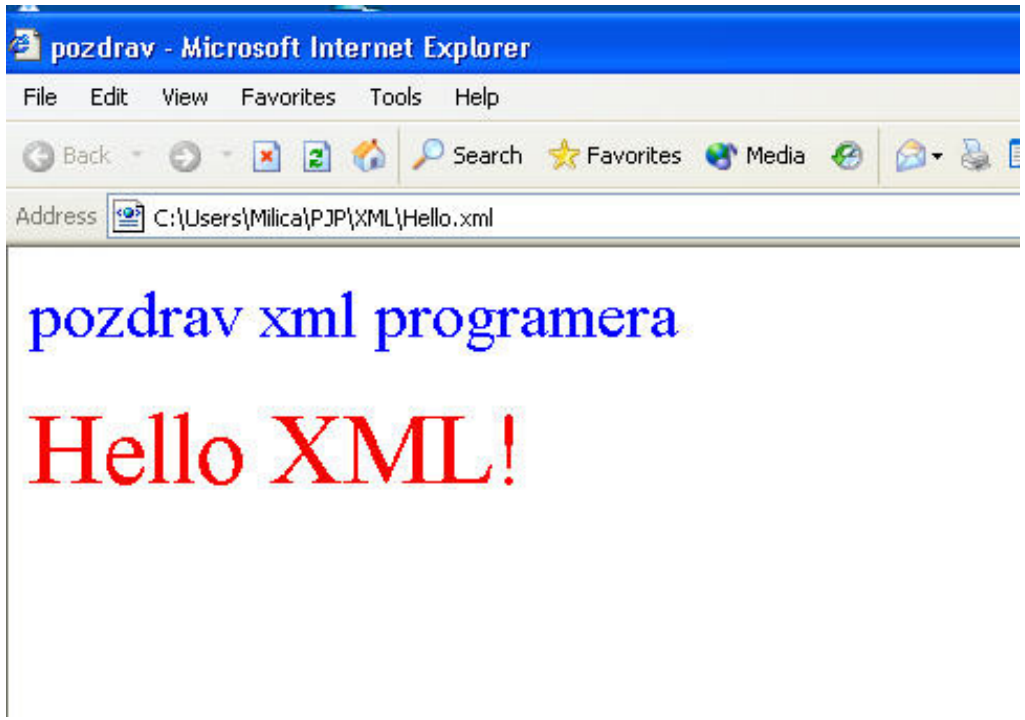
Željeni izlaz- HTML dokument

```
<html>
   <head>
        <title>pozdrav</title>
   </head>
   <body>
      <p>
        <font color="red"
        size="14">
            pozdrav xml programera
         </font>
      </p>
      <p>
            <font color="blue"
                size="16">
          Hello XML! </font>
      </p>
    </body>
</html>
```

XSLT Stylesheet dokument (pozdrav.xsl)

```
<?xml version="1.0" ?>
  <xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template match="/">
  <html>
     <head>
        <title>pozdrav</title>
     </head>
    <body>
       <p> <font>
                <xsl:attribute name="color">blue</xsl:attribute>
                <xsl:attribute name="size">6</xsl:attribute>
                 pozdrav xml programera
       </font> </p>
       <p> <font>
          <xsl:attribute name="color">red</xsl:attribute>
                  <xsl:attribute name="size">16</xsl:attribute>
          <xsl:value-of select="Pozdrav" />
       </font></p>
    </body>
  </html>
</xsl:template>
 </xsl:stylesheet>
```

Rezultat transformacije



Obrada XML dokumenata

XML parseri

XML parser

je softver koji čita XML dokument i čini dostupnim njegov sadržaj i strukturu aplikaciji preko API-a

API = Application Programming Interfaces

XML parseri

Postoje dve vrste parsera:

1) XML parseri koji verifikuju samo sintaksnu ispravnost XML dokumenta (da li je XML dokument dobro oformljen)

2) XML parseri vrše validaciju XML dokumenta u skladu sa XML šemom ili DTD

Modeli XML parsera

- **SAX** model = Simple API for XML
- **DOM** model = Document Object Model

Koraci obrade XML dokumenta

1) Parsiranje XML dokumenta (korišćenjem XML parsera)
   - Parser formira stablo čvorova (DOM),
   - Parser, za vreme parsiranja, šalje događaje aplikaciji (SAX)

2) Obrada dokumenta
   - Aplikacija pristupa i menja čvorove stabla korišćenjem interfejsa – DOM API
   - Aplikacija obrađuje poslate SAX događaje

3) Interpretacija parsiranog XML dokumenta u aplikaciji

DOM model (Document Object Model)

- Standardni objektno-orjentisani programski interfejs za obradu XML dokumenata
- W3C standard
- W3C DOM specifikacija pruža samo definiciju interfejsa za DOM biblioteke, a ne detalje njihove implementacije

DOM model

- DOM model reprezentuje XML dokument kao memorijsko stablo čvorova (DOM stablo) i omogućava, (preko skupa svojih interfejsa), *navigaciju* i *izmene* dokumenta

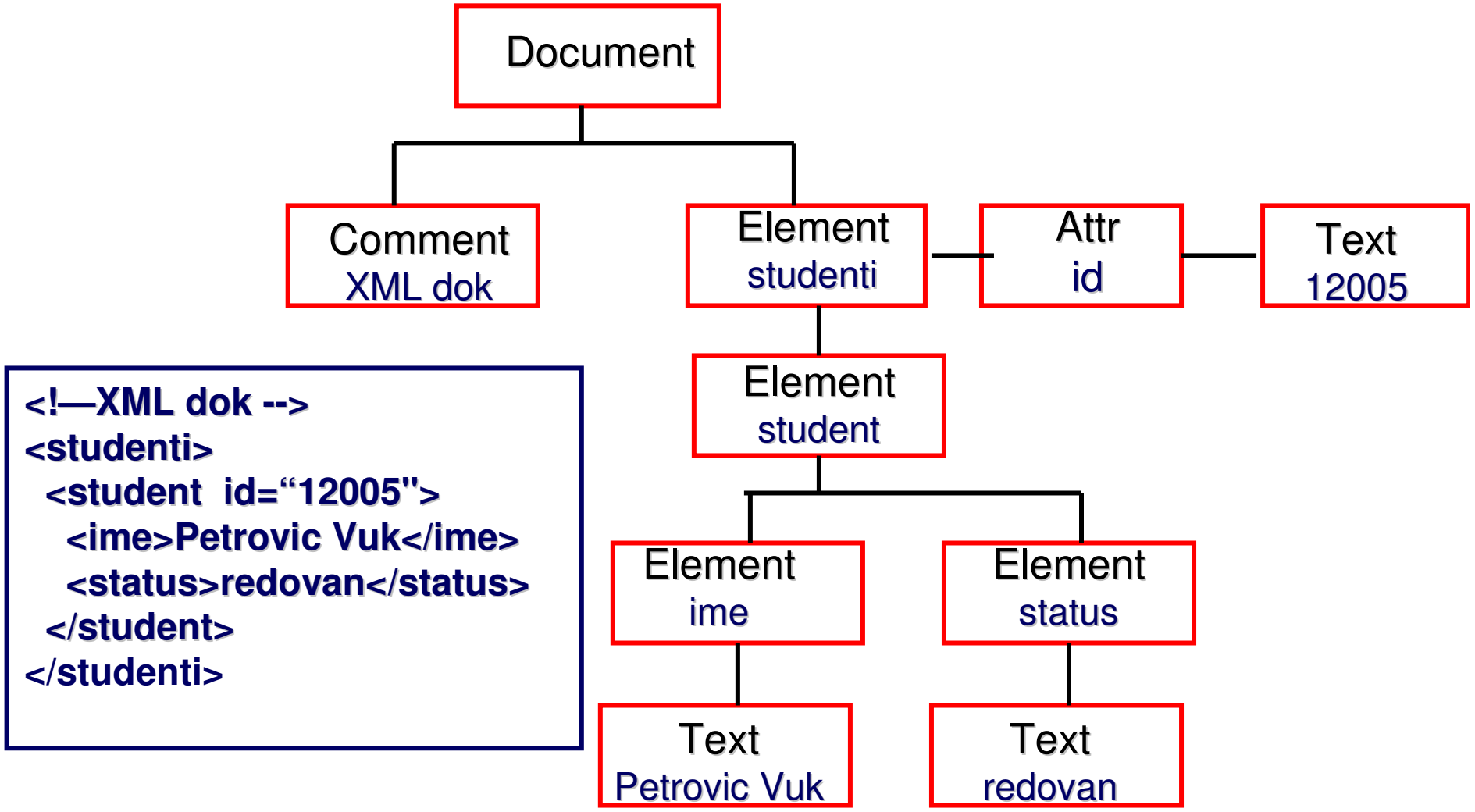
- Preko ovog modela se iz softverskih aplikacija može manipulisati sa XML dokumentima i njihovim elementima kao sa objektima

DOM model

- *Tipovi čvorova* u W3C specifikaciji:
  - Document
  - Element
  - Attribute
  - Character data
    - Text
    - Comment

- *Metode* za pristup i modifikaciju čvorova DOM stabla

DOM model

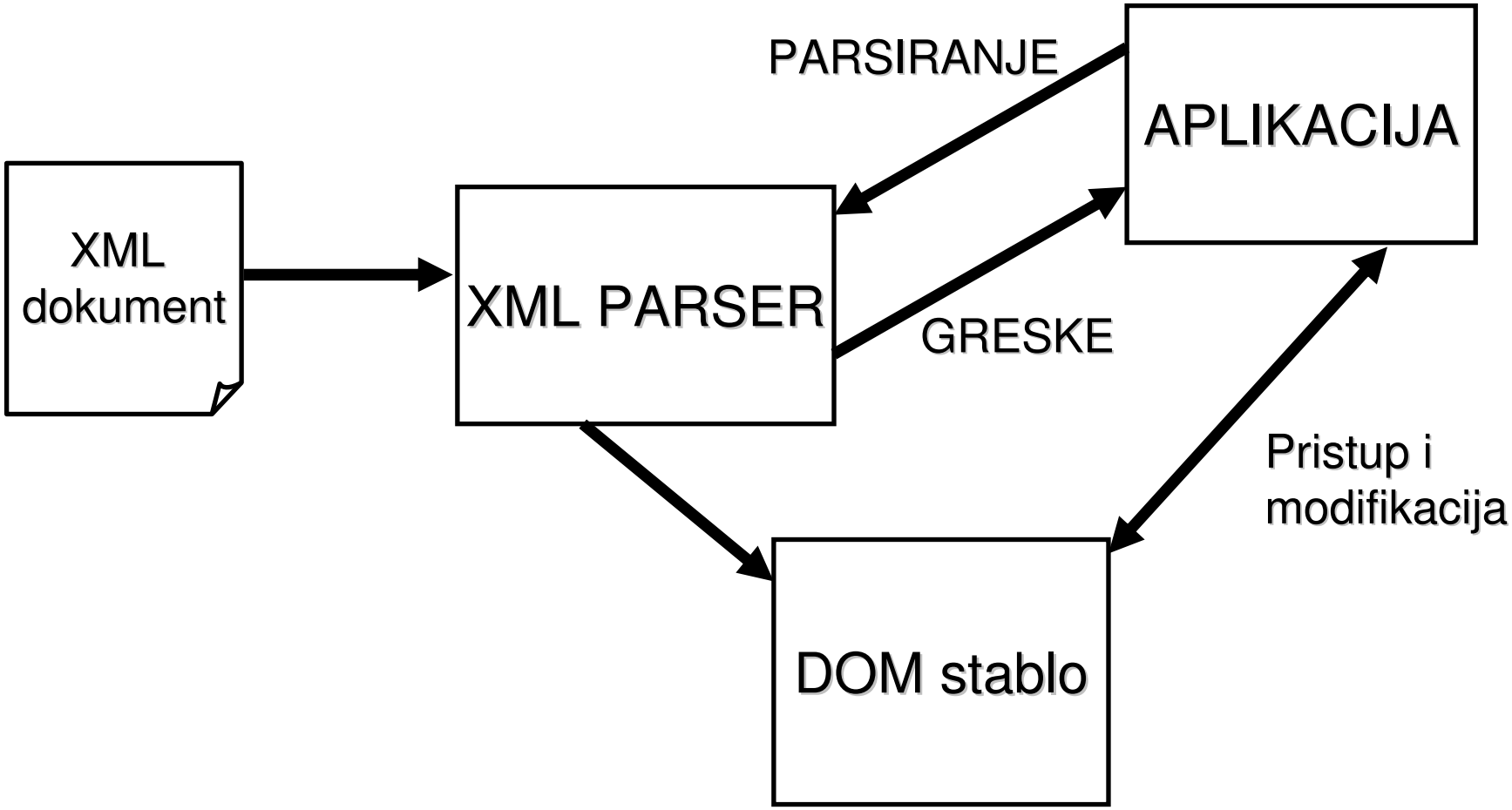

Document
Comment
XML dok
Element
studenti
Attr
id
Text
12005
Element
student
Element
ime
Element
status
Text
Petrovic Vuk
Text
redovan
<!—XML dok -->
<studenti>
<student id="12005">
<ime>Petrovic Vuk</ime>
<status>redovan</status>
</student>
</studenti>

Obrada XML dokumenta korišćenjem DOM parsera

DOM parser

- čita XML dokument od početka do kraja
- formira u memoriji strukturu stabla (DOM stablo) koja reprezentuje strukturu i sadržaj takvog dokumenta

Proces obrade XML dokumenata korišćenjem DOM parsera



Prednosti i nedostaci

- Prednosti
  - Dinamički pristup i modifikacija čvorova DOM stabla
  - Efikasno pretraživnje koje se zasniva na strukturi stabla
  - Isti interfejs za različte programske jezike (C++, Java, C#, ...)
- Nedostaci
  - Može da bude spor i zahteva dosta memorijskih resursa

Korišćenje DOM parsera

- Kada se zahteva obrada većine elemenata u XML dokumentu
- Kada se zahteva dinamički pristup i manipulacija sa XML dokumentom i njegovim elementima
- XML dokumenta sa složenom strukturom

Implementacija DOM modela

- DOM parseri u MS .NET-u:
  - XmlDocument
  - Microsoft XML Parser (za verziju 3.0 – MSXML 3.0)
  - Microsoft XML Core Services (za verziju 4.0 – MSXML 4.0)

    MSXML obezbeđuje još i sledeće servise:
    - XSD
    - XSLT 1.0
    - SAX
    - XPath

Implementacija DOM modela

- Java
  - JAXP Java API for XML Processing

    obezbeđuje sledeće servise:
    - DOM
    - SAX
    - XSLT

SAX model (Simple API for XML)

- *Event-based* model (zasnovan na događajima)
- Razvijen od strane XML-DEV grupe
- Industrijski standard

  Verzija 1.0 1998

  Verzija 2.0 2000

Obrada XML dokumenta korišćenjem SAX parsera

- SAX parser čita XML dokument kao stream
  - Prilikom čitanja XML dokumenta, SAX parser generiše dogadjaje kad god otkrije element/atribut/tekst/instrukciju obrade i šalje aplikaciji

- Aplikacija obrađuje događaje generisane od strane parsera
  - aplikacija implementira odgovarajuće hendlere koji sadrže metode kojima se ti dogadjaji obrađuju

Primer

```
<?xml version=”1.0”>
 <poruka>
    <pozdrav>Dobar dan!</pozdrav>
 </pozdrav>
```

Generisani SAX događaji:

start document

start element: poruka

start element: pozdrav

characters: Dobar dan!

end element: pozdrav

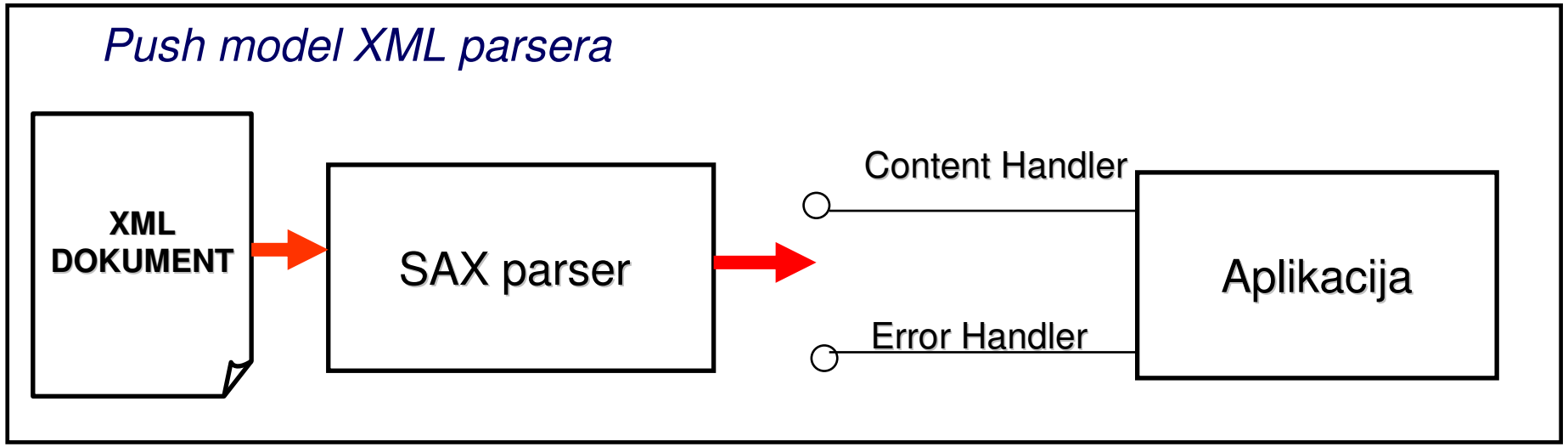
end element: poruka

end document

- sekvencijalan i
- “forward only” pristup

(svaki element se parsira naniže sve do listova pre nego što se pređe na sledeći element istog nivoa)

SAX – “push” model XML parsera



SAX parser

- Prednosti
  - Efikasan (veoma brz, ušteda memorije)
- Nedostaci
  - ne kreira memorijsko stablo za reprezentovanje XML dokumenta
  - sekvenicjalni pristup komponentama dokumenta

Korišćenje SAX parsera

- Kada se ne zahteva dinamički pristup i izmena elemenata XML dokumenta
- Obrada dokumenta sa prostom strukturom koja sadrže veliki obim podataka

Primeri SAX parsera

- **Xerces** (http://xml.apache.org)
- **Oracle XML Parser**
- **Project X** (Sun)
- **XML4J** (IBM)
- **MSXML 4.0** (Microsoft)

MS implementacija W3C XML standarda

.NET XML klase

MS implementacija W3C XML u .NET Framework-u

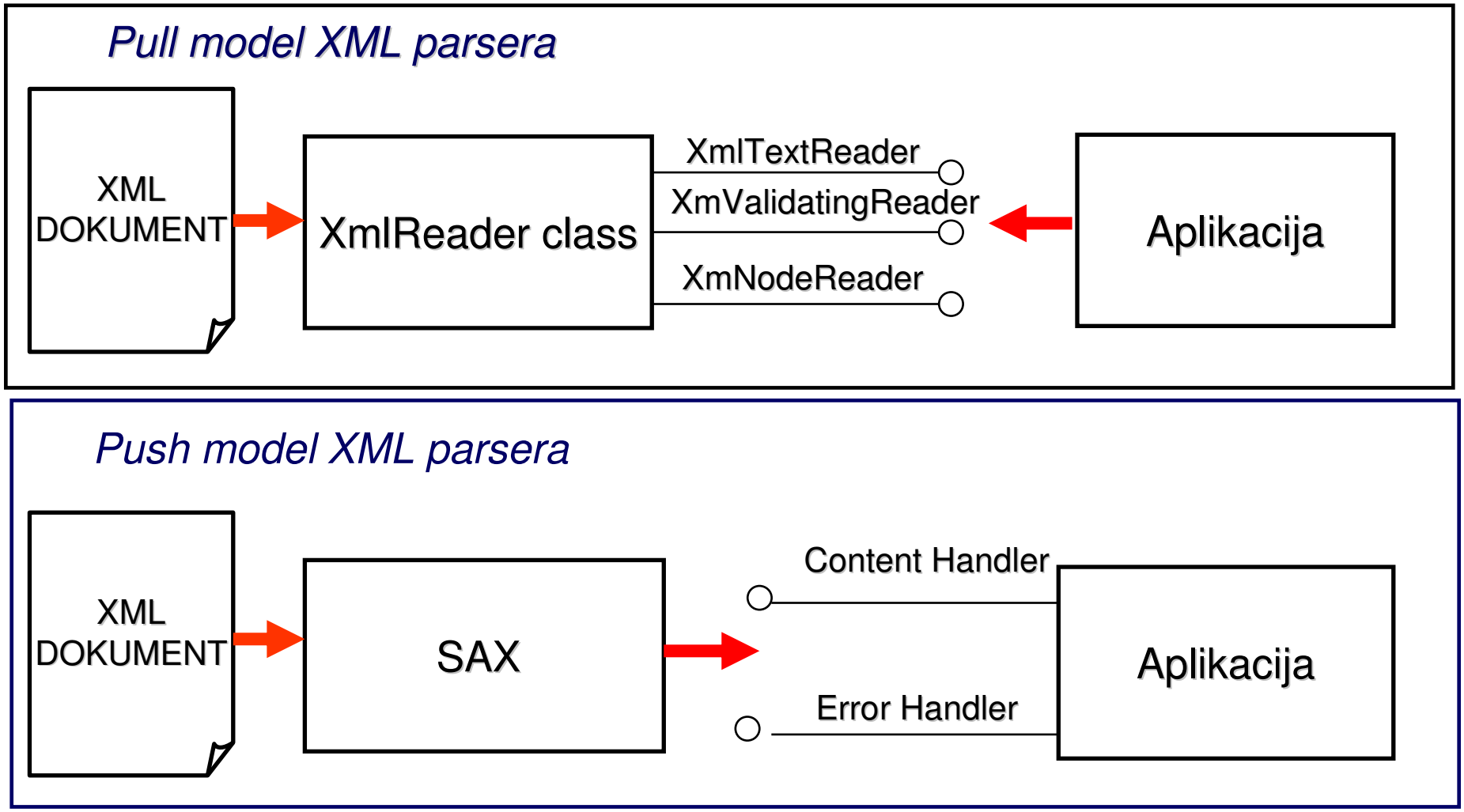
W3C XML Standard	.NET Framework namespace	.NET XML klase
XML 1.0	System.Xml	
XML Schema	System.Xml System.Xml.Schema	XmlSchema
XSLT	System.Xml.Xsl	XslTransform
XPath	System.Xml System.Xml.Path	Path
DOM	System.Xml	XmlDocument

Obrada XML dokumenata u .NET Framework-u

Postoje dve opcije obrade XML dokumenata:

- Preko **DOM**-a korišćenjem *XmlDocument* klase
- Preko **Pull** modela korišćenjem *XmlReader* klase.
  - Pull model je nov pristup za rad sa XML dokumentima

Pull&Push modeli parsera

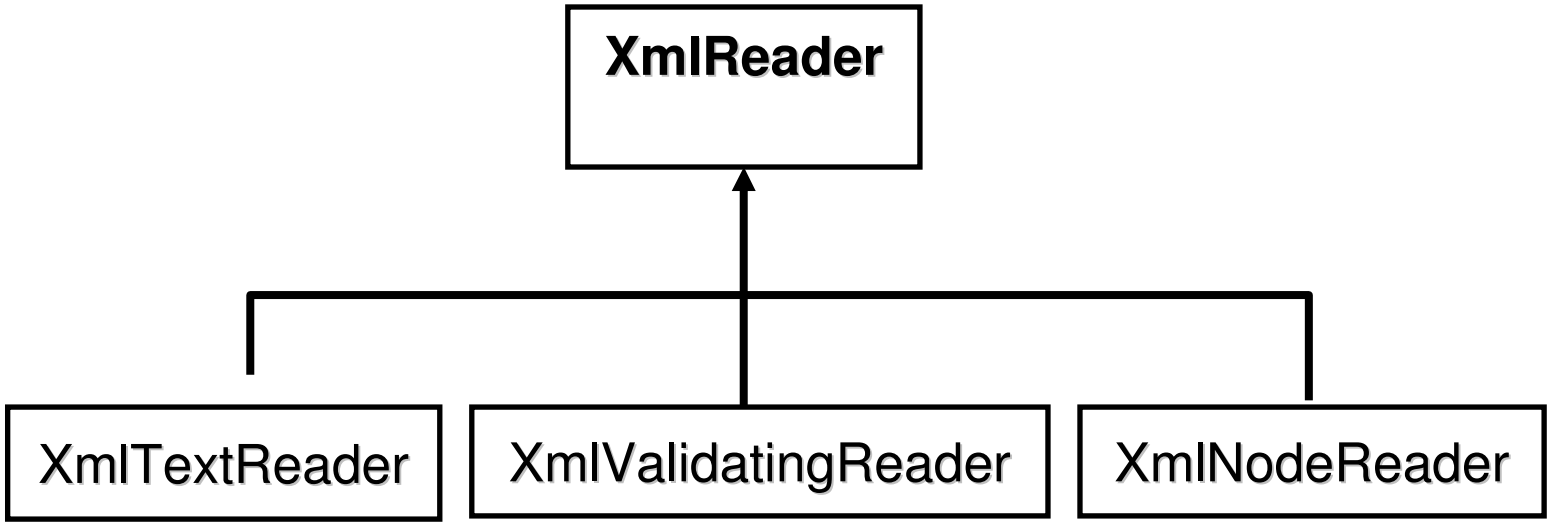


Pull&Push modeli parsera

- Pull model ne formira za XML dokument memorijsko stablo (slično kao i SAX)
- SAX je push model – dostavlja događaje aplikaciji koja ih obrađuje
- Pull model omogućava da aplikacija zahteva prolaženje kroz XML dokument i zatim selektovanje i pristup samo zahtevanim čvorovima
- Prednost Pull modela
  - Poboljšava performanse XmlReader-a

.NET XML klase: XmlReader klasa

- XmlReader je apstraktna klasa
- Reprezentuje pull model XML parsera
- Memorijski efikasan, forward-only, read-only pristup XML podacima



.NET klase: XmlTextReader klasa

- Provera da li je XML dokument dobro-oformljen
  - Ne proverava validnost
- Konstruktori omogućavaju čitanje XML iz različitih ulaznih izvora
  - datoteka, stream objekat ili TextReader
- *Read()* metoda omogućava navigaciju kroz čvorove XML dokumenta
  - obezbeđuje načine za čitanje sadržaja dokumenata, elemenata i atributa

.NET klase: XmlTextReader

Korišćenje XmlTextReader-a:

- Istanciranje XmlTextReader objekta
- Čitanje i obrada podataka
  - Parsira XML dokument korišćenjem *Read()* metode u *While* petlji

.NET XML klase: xmlTextReader

```
// konstruktor
XmlTextReader reader = new XmlTextReader ("Studenti.xml");
while (reader.Read())
{
// obrada tekućeg čvora
        switch (reader.NodeType)
        {
                case XmlNodeType.Element :
                        Console.Write("<" + reader.Name +  ">");
                        break;
                case XmlNodeType.EndElement:
                        Console.Write("</" +  reader.Name + ">");
                        break;
                case XmlNodeType.Text :
                        Console.Write(reader.Value);
                        break;
                default:Console.WriteLine();break;
        }
}
reader.Close ();
```

XmlValidatingReader& XmlNodeReader

- **XmlValidatingReader** obezbeđuje podršku za *validaciju* XML dokumenta u skladu sa DTD ili XSD

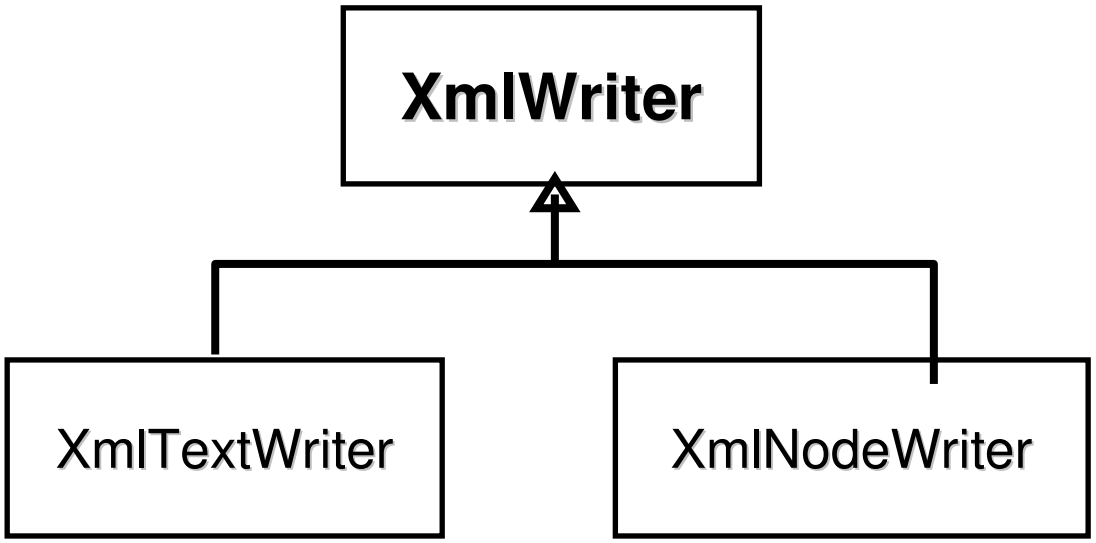
- **XmlNodeReader** omogućava čitanje podstabla XML DOM stabla
  - Ne podržava validaciju

.NET XML: klase: XmlWriter

Programsko generisanje XML dokumenata u datoteku, stream, TextWriter

.NET XML: klase: XmlWriter

- XMLWriter je apstraktna klasa
- Reprezentuje brzi, forward-only, memorijski efikasan XML writer



.NET klase: XmlTextWriter

Korišćenje XmlTextWriter-a:

1) Istanciranje XmlTextWriter-a (konstruktor)
2) Postavljanje property-a (za formatiranje itd.)
3) Izvršavanje *Write* metode za generisanje XML
4) Izvršavanje *close()* metode

Primer

```
//konstruktor
XmlTextWriter writer = new XmlTextWriter
("knjiga.xml",System.Text .Encoding .UTF8 );
//kreiranje root elementa
writer.WriteStartElement ("Knjiga");
//kreiranje elementa Naslov
writer.WriteStartElement ("Naslov");
writer.WriteString ("Profesional ASP.NET");
writer.WriteEndElement();
//kreiranje elementa Autor
writer.WriteStartElement ("Autor");
writer.WriteString ("Richard Anderson");
writer.WriteEndElement();
//kreiranje elementa ISBN
writer.WriteStartElement("ISBN");
writer.WriteString ("86-7991-155-0");
writer.WriteEndElement();
//kreiranje elementa Datum
writer.WriteStartElement("Datum");
writer.WriteString ("Jun 2002");
writer.WriteEndElement();
//kreiranje elementa Izdavac
writer.WriteStartElement("Izdavac");
writer.WriteString ("Wrox Press, Birmingham");
writer.WriteEndElement();
//zatvaranje root elementa
writer.WriteEndElement ();
writer.Flush ();
writer.Close ();
```

.NET XML klase: XmlDocument

- XmlDocument obezbedjuje podršku W3C DOM modela
  - Reprezentuje XML dokument kao memorijsku strukturu stabla
  - Metode:

    Load()

    Save()

- Izvedena iz XmlNode klase

.NET klase: XmlDocument

- Korišćenje XmlDocument klase

```
// konstruktor
XmlDocument doc = new XmlDocument ();
// ucitavanje XML dokumenta
doc.Load ("Studenti.xml");
// ispis sadržaj xml dokumenta
Console.WriteLine(doc.InnerXml.ToString ());
```

.NET Klase: XslTransform klasa

- System.Xml.Xsl namespace
- transformiše ulazni XML dokument korišćenjem XSLT stylesheet-a
- Ključne metode
  - *Load*
  - *Transform*

.NET Klase: XslTransform klasa

```
// kreiranje XslTransform objekta
XslTransfrom transformacija = new XslTransform;
// punjenje stylesheet doc
transformacija.Load("pozdrav.xsl");
// transformacija
transformacija.Transform("pozdrav.xml",
   "pozdrav.html");
```

.NET Klase: XslTransform klasa

```
string XmlPath = “Hello.xml";
string ResultHtmlPath = “Hello.html";
XmlTextReader xslt = new
XmlTextReader(“Hello.xst");
XslTransform transformacija = new
XslTransform();
transformacija.Load(xslt);
transformacija.Transform(XmlPath,
ResultHtmlPath);
```