

XML-Datenmodellierung-DTD

```
<!ELEMENT Firma (Abteilungen, Projekte?)>
```

```
<!ELEMENT Abteilungen (Abteilung*)>
```

```
<!ELEMENT Abteilung (Name, Unterabteilungen?)>
```

```
<!ATTLIST Abteilung
```

```
    id ID #REQUIRED
```

```
    projekte IDREFS #IMPLIED>
```

```
<!ELEMENT Name (#PCDATA)>
```

```
<!ELEMENT Unterabteilungen (Abteilung+)>
```

```
<!ELEMENT Projekte (Projekt*)>
```

```
<!ELEMENT Projekt (Titel, Budget)>
```

```
<!ATTLIST Projekt
```

```
    id ID #REQUIRED
```

```
    abteilungen IDREFS #IMPLIED>
```

```
<!ELEMENT Titel (#PCDATA)>
```

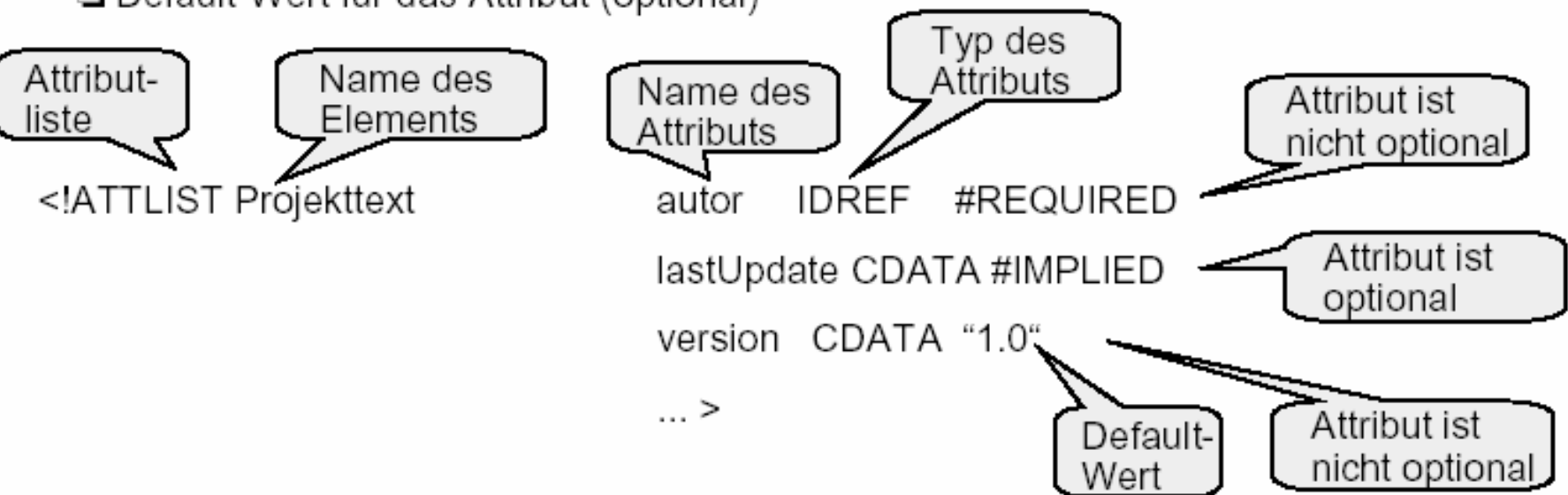
```
<!ELEMENT Budget (#PCDATA)>
```

Attribute und Attributdeklaration

Deklaration einer Liste von Attributen zu einem Element in der DTD.

Die einzelne Attributdefinition besteht aus

- ❑ Name des Attributs
- ❑ Typ des Attributs (siehe nächste Folie)
- ❑ Angabe, ob das Attribut optional ist (#REQUIRED, #IMPLIED)
- ❑ Default-Wert für das Attribut (optional)



Darstellung von Beziehungen (1:N)

1:N Beziehungen (Hierarchien):

Schachtelung von XML-Elementen in andere XML-Elemente

Beispiel (Beziehung Abteilung-Unterabteilungen)

```
<!ELEMENT Abteilung (Name, Unterabteilungen?)  
<!ELEMENT Unterabteilungen (Abteilung*)>
```

In der DTD

```
<Abteilung id="LTSW">  
  <Name>Leitung Software</Name>  
  <Unterabteilungen>  
    <Abteilung id="UXSW">  
      <Name>Unix Software</Name>  
    </Abteilung>  
    <Abteilung id="MFSW">  
      <Name>Mainframe Software</Name>  
    </Abteilung>  
  </Unterabteilungen>  
  ...  
</Abteilung>
```

Im
XML Dokument

Darstellung von Beziehungen (N:M)

Darstellung über XML-Attribute

Voraussetzung: Beide Elemente sind Teil des selben Dokuments

Beispiel (Darstellung einer N:M-Beziehung in beiden Richtungen)

```
<!ATTLIST Abteilung    id ID                #REQUIRED  
                        projekte IDREFS      #IMPLIED>
```

In der DTD

```
<!ATTLIST Projekt      id ID                #REQUIRED  
                        abteilungen IDREFS    #IMPLIED>
```

```
<Abteilung id = "UXSW" projekte = "P100 P200"> ... </Abteilung>
```

Im XML
Dokument

```
<Projekt id = "P100" abteilungen = "UXSW MFSW LTSW" > ...</Projekt>
```

Eindeutigkeit der Identifikatoren und Existenz von referenzierten Identifikatoren (referentielle Integrität) werden vom XML Parser überprüft.

Alternativen zur Darstellung von Beziehungen

a) Modellierungsalternative: Verwendung separater Elemente


```
<Projekt id="P100">
  <Titel>DB Fahrplaene</Titel>
  <Budget>300000</Budget>
  <Betreuung Abteilung="MFSW"/>
  <Betreuung Abteilung="UXSW"/>
</Projekt>
```

```
<!ELEMENT Projekt (Titel, Budget, Betreuung*)>
<!ELEMENT Betreuung EMPTY>
<!ATTLIST Betreuung Abteilung IDREF #REQUIRED>
```

b) Verwendung des allgemeinen Link-Mechanismus in XML

(bei Beziehung zwischen Elementen in verschiedenen Dokumenten)

```
<Projekt id="P100">
  <Titel>DB Fahrplaene</Titel>
  <Budget>300000</Budget>
  <Betreuung xlink:type="simple" xlink:href="http://.../abteilungen.xml|MFSW" />
  <Betreuung xlink:type="simple" xlink:href="http://.../abteilungen.xml|UXSW" />
</Projekt>
```



Element ID
Verweis

IDs and IDREFs

An element can have at most one attribute of type ID

The ID attribute value of each element in an XML document must be distinct

→ ID attribute (value) is an object identifier

An attribute of type IDREF must contain the ID value of an element in the same document

An attribute of type IDREFS contains a set of (0 or more) ID values. Each ID value must contain the ID value of an element in the same document

IDs and IDREFs are untyped, unfortunately

- Example below: The *owners* attribute of an account may contain a reference to another account, which is meaningless;
owners attribute should ideally be constrained to refer to customer elements

Example: Bank XML

```
<!DOCTYPE bank [  
  <! ELEMENT bank ( ( account | customer | depositor)+ )>  
  <! ELEMENT account (account-number branch-name balance)>  
  <! ELEMENT customer(customer-name customer-street  
                                customer-city)>  
  <! ELEMENT depositor (customer-name account-number)>  
  <! ELEMENT account-number (#PCDATA)>  
  <! ELEMENT branch-name (#PCDATA)>  
  <! ELEMENT balance(#PCDATA)>  
  <! ELEMENT customer-name(#PCDATA)>  
  <! ELEMENT customer-street(#PCDATA)>  
  <! ELEMENT customer-city(#PCDATA)>  
>
```

Example: Bank DTD

Bank DTD with ID and IDREF attribute types

```
<!DOCTYPE bank-2[
  <!ELEMENT account (branch-name, balance)>
  <!ATTLIST account
    account-number ID #REQUIRED
    owners IDREFS #REQUIRED>
  <!ELEMENT customer(customer-name, customer-street,
    customer-city)>
  <!ATTLIST customer
    customer-id ID #REQUIRED
    accounts IDREFS #REQUIRED>
  ... declarations for branch, balance, customer-name,
    customer-street and customer-city
]>
```


Example: Bank XML (ID,IDREF)

<bank-2>

```
<account account-number="A-401" owners="C100 C102">  
  <branch-name> Downtown </branch-name>  
  <balance>500 </balance>  
</account>
```

...

```
<customer customer-id="C100" accounts="A-401">  
  <customer-name>Joe</customer-name>  
  <customer-street>Monroe</customer-street>  
  <customer-city>Madison</customer-city>  
</customer>
```

```
<customer customer-id="C102" accounts="A-401 A-402">  
  <customer-name> Mary</customer-name>  
  <customer-street> Erin</customer-street>  
  <customer-city> Newark </customer-city>  
</customer>
```

</bank-2>

XML Schema Version of Bank DTD

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.banks.org"
            xmlns="http://www.banks.org" >
<xsd:element name="bank" type="BankType"/>
<xsd:element name="account">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="account-number" type="xsd:string"/>
      <xsd:element name="branch-name" type="xsd:string"/>
      <xsd:element name="balance" type="xsd:decimal"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>          .... definitions of customer and depositor ....
<xsd:complexType name="BankType">
  <xsd:choice minOccurs="1" maxOccurs="unbounded">
    <xsd:element ref="account"/>
    <xsd:element ref="customer"/>
    <xsd:element ref="depositor"/>
  </xsd:choice>
</xsd:complexType>
</xsd:schema>
```