Webbasierte Anwendungen: Verteilte Systeme

Dokumentation der ersten Phase

von Christoph Czisch (mi1014)

Aufgabe 1:

Erklären Sie kurz die Begriffe Wohlgeformtheit, Validität und Namespaces im Bezug auf XML und XML-Schema.

Wohlgeformtheit:
Ein Begriff der Dokumente beschreibt die den Syntaxregeln für XML entsprechen. Ein wohlgeformtes XML Dokument verfügt über Tags mit den richtigen Begrenzer, folgt ein End-Tag auf ein Start-Tag und die Elemente überlappen sich nicht.

Validität:
Die Validität (Gültigkeit) geht noch über die Wohlgeformtheit hinaus. Möchte man ein spezielles Programm davor schützen das z.b. unerwartete Elementtypen das Programm negativ beeinflussen braucht man ein formelles Dokumentmodell. Bei einem Dokument-modell handelt es sich um eine Art Blaupause für eine Instanz einer Markup-Sprache - z.B. wann darf welcher Elementtyp verwendet werden. Bei einem mathematischen Programm das nur zwei zahlen mit einander addieren soll könnte man vorgeben das es sich bei diesen beiden Zahlen ausschließlich um Integer handeln darf. Würde man nun versuchen zwei floats mit einander zu addieren bekäme man kein Ergebnis bzw eine entsprechende Warnmeldung.
Entspricht eine Dokumentinstanz einem Dokumentmodell spricht man von Validität. Zwei Arten um Markup-sprache zu formalisieren sind DTD (Document Type Definition)
und Schemas.

Namespaces (Namensräume):
-mechanismus über den Elemente und Attribute Gruppen zu gewiesen werden können
-es wird benutzt um das Vokabular eines xml-dokumentes eindeutig zu identifizieren dabei könne auch mehrere XML-Sprachen in einem Dokument gemischt werden.
-Namensräume werden durch URIs dargestellt
-zusätzlich existiert noch ein Präfix Mechanismus. werden mehrere namensräume benutzt können diese durch ein Präfix unterschieden werden. hierbei ist zu beachten, dass ein Kindelement eines Elements mit Präfix nicht automatisch den selben namensraum erben, es muss also mit dem selben bzw einem entsprechenden Präfix versehen werden.

Aufgabe 2:

a) Erzeugen Sie ein XML-Dokument, dass die Daten des folgenden Formulars vollständig erfasst:

http://www.gm.fh-koeln.de/~vsch/anmeldung/gruppenanmeldung.html

Um diese Aufgabe zu lösen habe ich mir zunächst die Frage gestellt wie die Struktur von so einer Anmeldung rein konzeptionell aussehen sollte. Folgende Punkte erschienen mir wichtig:

- es sollen sich Gruppen eintragen können
- der Gruppenleiter trägt sich und seine ganze Gruppe ein
- es sollten ids für alle Gruppen vergeben werden

Das Ergebnis sah dann folgendermaßen aus:

```xml
<anmeldung>
    <gruppe id="1">
        <gruppenleiter>
            <vorname>Paul</vorname>
            <nachname>Mustermann</nachname>
            <email>muster@mann.de</email>
            <geburtsdatum>01.01.1985</geburtsdatum>
            <erfahrung>fortgeschritten</erfahrung>
            <schlagzeug>vorhanden</schlagzeug>
            <anmerkung>Keine Anmerkung</anmerkung>
        </gruppenleiter>
        <gruppenmitgliegder>
            <person>
                <vorname>Bernd</vorname>
                <nachname>Schulze</nachname>
                <email>Bernd@schulze.de</email>
                <geburtsdatum>24.06.1984</geburtsdatum>
                <erfahrung>fortgeschritten</erfahrung>
                <schlagzeug> nicht vorhanden</schlagzeug>
            </person>
            <person>
                <vorname>Claudia</vorname>
                <nachname>Müller</nachname>
                <email>claudia@mueller.de</email>
                <geburtsdatum>24.08.1986</geburtsdatum>
                <erfahrung>fortgeschritten</erfahrung>
                <schlagzeug>vorhanden</schlagzeug>
            </person>
        </gruppenmitgliegder>
    </gruppe>
</anmeldung>
```

Das erstellte XML Dokument besteht aus einem Wurzelelement <anmeldung>.
Weiter besteht es aus einem Array mit ID für die Gruppen <gruppe id="1"> um die einzelnen Gruppen in einer seperaten Gruppen-XML auszulagern. Der Gruppenleiter, der auch seine ganze Gruppe einträgt wird unter <gruppenleiter> mit Vor-, Nachname, Email, Geburtsdatum, Erfahrung, Schlazeug vorhandensein und spezial nur für den Gruppenleiter der Möglichkeit eine Anmerkung zu hinterlassen erfasst.
Ein weiteres Array wurde für die restlichen Gruppenmitglieder <person> angelegt in dem die einelnen Gruppenmitglieder unter <person> mit Vor-, Nachname, Email, Geburtsdatum, Erfahrung, Schlazeug vorhandensein erfasst werden.

b) Erzeugen Sie ein JSON-Dokument, dass zu ihrem XML-Dokument äquivalent ist.

Hier wurde ein json Dokument erstellt und hier sind die Arrays für <gruppe> und <person>.

```
{
    "anmeldung":{
        "gruppe": [
            {
            "id": "1",
            "gruppenleiter":{
                "vorname":"Paul",
                "nachname":"Mustermann",
                "email":"muster@mann.de",
                "geburtsdatum":"01.01.1985",
                "erfahrung":"fortgeschritten",
                "schlagzeug":"vorhanden",
                "anmerkung":"Keine Anmerkung"
            },
            "gruppenmitglieder": {
            "person":[
                {
                    "vorname":"Bernd",
                    "nachname":"Schulze",
                    "email":"Bernd@schulze.de",
                    "geburtsdatum":"24.06.1984",
                    "erfahrung":"fortgeschritten",
                    "schlagzeug":"nicht vorhanden"
                },
                {
                    "vorname":"Claudia",
                    "nachname":"Müller",
                    "email":"claudia@mueller.de",
                    "geburtsdatum":"24.08.1986",
                    "erfahrung":"fortgeschritten",
                    "schlagzeug":"vorhanden"
                }
            ]
            }
        }
    ]
```

Aufgabe 3:

a) Gegeben ist folgendes Rezept:

http://www.chefkoch.de/rezepte/24641006006067/Lenchen-s-Schokoladenkuchen.html

Entwickeln Sie ein XML-Dokument, in dem die Daten des Rezeptes abgebildet werden. Achten Sie darauf, dass das Dokument semantisch möglichst reichhaltig ist. Bei dieser und den folgenden Aufgaben lassen sie bitte die Daten in der Marginalspalte auf der rechten Seite weg.

Zunächst habe ich die Seite versucht von Oben nach Unten in einem XML Dokument zu erfassen. Die folgenden Elemente habe ich hierzu erstellt.

<rezept> (Wurzelelement>, <titel>, <fotos>, <zutaten> (mit Kindelementen), <portionen>, <zubereitung> (mit Kindelementen) und <kommentare>

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rezept xmlns="http://example.org/Rezepte" xmlns:xsi="http
    <titel>Lenchen's Schokoladenkuchen</titel>
    <fotos></fotos>
    <zutaten>..
    <portionen>16</portionen>
    <zubereitung>..
    <kommentare>..
</rezept>
```

Im weitern werde ich auf die jeweiligen Bereiche näher eingehen:

```
<zutaten>
    <zutat id="1">
        <bezeichnung>Butter</bezeichnung>
        <menge>200.0</menge>
        <einheit>g</einheit>
    </zutat>
    <zutat id="2">
```

Für jede Zutat habe ich ein Element <zutat> mit ID angelegt (falls mal auf diese zutaten zu gegriffen werden muss) und in dieses <zutat> Element habe ich 3 wetiere Elemente verschachtelt <bezeichnung>, <mege> und <einheit>

Der Bereich Zubereitung wurde wie folgt gestaltet:

```
  <zubereitung>
      <arbeitszeit>60</arbeitszeit>
      <schwierigkeitsgrad>simpel</schwierigkeitsgrad>
      <brennwertAnzeige>
          <menge2>295</menge2>
          <brennwert>kcal</brennwert>
      </brennwertAnzeige>
      <text>Butter und Schokolade im Wasserbad schmelzen. Eier trenne
i 180°Grad 40 - 50 Minuten backen.</text>
  </zubereitung>
```

Und die Kommentare wurder wie folgt gestaltet:

```
<kommentare>
    <eintrag>
        <autor>swieselchen</autor>
        <datum>2002-02-07T18:49:00</datum>
        <text>Habe Deinen Kuchen gestern gebacken (kleine Abwandlung: s
    </text>
    </eintrag>
    <eintrag>
        <autor>thecook</autor>
        <datum>2006-12-07T12:24:00</datum>
        <text>Habe ihn eben gebacken und bin begeistert.Super Rezept:-)
    </text>
    </eintrag>
    <eintrag>
        <autor>2</autor>
        <datum>2013-04-13T16:14:03.402+02:00</datum>
        <text>hdskjh</text>
    </eintrag>
</kommentare>
```

Hervorzuheben wäre and ieser Stelle noch das Element <datum> und insbesondere desen Schreibweise. Da ich im nachfolgenden XML-Schema den Typ dateTime verwende musste ich mich an diese Schreibweise halten yy-mm-ddThh:mm:ss

b) Betrachten Sie nun andere Rezepte auf der Webseite http://www.chefkoch.de. Beschreiben Sie welche Gemeinsamkeiten die Rezepte hinsichtlich ihrer Daten haben und worin Sie sich unterscheiden.

Ich konnte bei meiner Recherche keine Unterschiede hinsichtlich der Daten/Struktur festel-

len. Die Gemeinsamkeiten sind:

- der Aufbau von Rezepten: Titel > Bild > Zutaten > Portionen > zubereitung > Kommentare
- Kommentare sind Optional
- midestens: 1 Zutat, ein Bild,
- Immer eine Zubereitungsanweiisung

c) + d)

Arbeiten Sie die Kriterien heraus, die für die Entwicklung einer XML-Schema-Datei beachtet werden müssen. Die Schema-Datei soll die Struktur für eine XML-Datei definieren, in der mehrere unterschiedliche Rezepte gespeichert werden können.
Ziel ist es, dass das XML-Schema möglichst restriktiv ist, so dass in der XML-Datei möglichst semantisch sinnvolle Daten bezüglich der Rezepte gespeichert werden können. Ziehen Sie beim Aufstellen der Kriterien u.A. folgende Fragestellungen in Betracht:

Welche Daten müssen in simple und welche in complex-types abgebildet werden?
Für welche Daten ist die Abbildung in Attributen sinnvoller?
Welche Datentypen müssen für die Elemente definiert werden?
Welche Restriktionen müssen definiert werden?

Erstellen Sie nun ein XML-Schema auf Basis ihrer zuvor definierten Kriterien. Generieren Sie nun auf Basis des Schemas eine XML-Datei und füllen Sie diese mit zwei unterschiedlichen und validen Datensätzen.

- Es muss viel Text gespeichert werden, z.B. Zubereitung, dafür eignet sich string.

- Bei den Foto erwartet man einen Link daher eignet sich hier anyURI als Typ

- Zutaten; hier handelt es sich um einen complexType, also einen Type der sich aus anderen meist simpleTypes zusammensetzt. Eine Zutat kann beliebig oft aber mindestens einmal vorkommen - ein Rezept ohne Zutat ist ja kein Rezept - daher braucht mein Element noch zwei Attribute <xs:element name="zutat" maxOccurs="unbounded" minOccurs="1"> . Eine weitere besonderheit ist hier noch zu beachten - und zwar im Zusammenhang mit den Einheiten bei Zutaten (g, Kg, usw.). Da die FAQ von chefkoch.de keine Vorgaben machen wie die Einheiten auszusehen haben - es wird lediglich eine Empfehlung gemacht. Habe ich an dieser Stelle eine Restriktion vorgesehen und als Beispiel auf einige Einheiten beschränkt - diese Leiste kann man natürlich vorsetzen. So wird eine einheitliche Einheitenbenennung gewährleistet - schreibweisen in Rezepten wie: Teeloeffel, tL,Tl,tl werden so durch eine begrenzte aber vorgegebene auswahl beschränkt (TL).

-da ich an manchen stellen im XML mit IDs arbeite, ich habe mich bei der Vergabe von IDs auf Metadaten beschränkt da ich Element auch als solche abbilden will. D.h. 200 g sind für mich zwei Elemente nämlich 200 = <menge> und g = <einheit> UND IDs sind für mich

dementsprechend Metadaten

- Eine weitere Restriktion habe ich bei den Schwierigkeitsgraden der Rezepte vorgesehen. Hier kann unter simpel, mittel und schwierig ausgewählt werden.

-Da Kommentare optional sind habe ich das Element
<xs:element name="eintrag" maxOccurs="unbounded" minOccurs="0"> mit einem Attribut minOccurs="0" versehen

Alles weiter ist ohne große Erklärung aus dem Code zu entnehmen.

Aufgabe 4:

Ich verzichte an dieser Stelle auf eine lange und ermüdende Erläuterung. Im Code habe ich alles notwendige kommentiert.

Aufabe 5:

Webkommunikation ermöglicht eine Vielzahl von Verbindungstechnologien. Eine einmal erstellte Seite kann so mit Hilfe von Standards wie z.b. HTML beliebig oft von verschiedenen Empfängern angezeigt werden. Um nun Informationen und Daten in strukturierterweise wiedergeben und speichern zu können eignen sich Technologien wie XML bzw JSON, die einen schnellen und unabhängigen Datenaustausch gewährleisten.

Nehmen wir als Beispiel eine Kochrezepte-webseite dann ist es effektiver das Rezept mit all seinen Bestandteilen wie z.b. zutaten, Bildern usw. formal zu definieren und in einer separaten (XML/JSON)-Datei zu speichern aus der man sich dann bedient wenn man z.b. eine Quickinfo für diese Rezept anzeigen will. Diese könnte dann aus Titel und Bild bestehen. Klickt man diese Quickinfo an und gelangt zum eigentlichen Rezept werden die Daten aus der Gleichen Datei verwendet werden.

Mit Hilfe von XML / JSON können so Dokumente angelegt und erweitert werden die bestimmten Vorgaben folgen z.b. Mengenangaben dürfen nur als Integer gespeichert werden. So kann meist problemlos Programm- und Plattformunabhängig auf die Daten zugegriffen bzw diese Daten manipuliert werden.

vor und Nachteile

json:
- Durch JavaScript Syntax ist ein json Dokument gleichzeitig ein ausführbares JavaScript und per eval() interpretierbar
-Datenaustauschformat
-Keine Trennung von Daten und Code
-durch einfache Schreibweise ist json für Menschen leicht verständlich
-Beliebtes Einsatzgebiet von json ist die asynchrone Übertragung von Daten zwischen Server und WebSeiten
-kleinerer Overhead
-unterstützt nur einige grundlegende Datentypen

- boolescher Wert
- Nullwert
- Zeichenkette
- Array
- Zahl
- Objekt

xml:
-Programmiersprachen- und Plattformunabhängig durch Trennung von Inhalt und Darstellung
-Formale Überprüfung des Dokumentes erfolgt durch DTD bzw. XML-Schemas und nicht in der Anwendung
-offener Standard
-Werte und Eigenschaften können sowohl als Attribute als auch Kindknoten beschrieben werden, kann zu Problemen führen wenn keine strikte Spezifizierung vorhanden ist
-durch Formalisierung und Einschränkungen mittels XML-Schemas können XML Dateien genau an die jeweiligen „Bedürfnisse“ angepasst werden.
-kann unter anderem für die Beschreibung von:

- gewöhnlichen Dokumenten
- strukturierten Datensätzen
- Metadaten für Webseiten
- grafische Darstellung genutzt werden