PI.SWA.SA.VO Software Architekturen und Web Technologien

1. Termin WS 2007
Schriftliche Einzelprüfung
Fr 25.01.2008

Prüfer: Derntl/Motschnig

Anweisungen:

- **Arbeitszeit: 90 Minuten**, es sind **max. 90 Punkte** zu erreichen (die Anzahl der Punkte, die pro Aufgabe zu erreichen ist, finden Sie jeweils in eckiger Klammer). Kalkulieren Sie also für jeden erreichbaren Punkt ca. eine Minute Arbeitszeit.
- Halten Sie einen **Lichtbildausweis** (Studentenausweis, Führerschein, Pass, etc.) bereit.
- Lösen Sie die Aufgaben direkt auf den Angabeblättern und schreiben Sie **leserlich**.
- Falls Sie zusätzliche Blätter benötigen, fragen Sie die Prüfungsaufsicht.
- Es sind **keine Unterlagen**, Handies, PDA, Laptops, Taschenrechner, Headsets, oder sonstige Hilfsmittel erlaubt!

Hinweise:

- Ergebnisse werden bis spätestens Di 12.2. im PISWI verfügbar sein.
- Einsicht ist Mi 13.2., 10-11 Uhr, im Dienstzimmer Derntl, Rathausstrasse 19, 1. Stock, Tür 9
- Dies ist der erste von insgesamt vier Terminen aus dem Stoffsemester WS 2007. Drei weitere Termine folgen im SS 2008.

Aufgabe 1: XML Familie [35]

Gegeben ist folgendes XML Schema. (Hinweis: die Zahlen am Beginn jeder Zeile sind Zeilennummern, auf die Sie sich bei der Lösung der Aufgabe beziehen können.)

```
 1 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 2
 3  <xsd:complexType name="autos_t">
 4   <xsd:sequence minOccurs="2" maxOccurs="unbounded">
 5    <xsd:element name="auto" type="auto_t" minOccurs="0"/>
 6    <xsd:element name="haendler" type="xsd:string" minOccurs="0"/>
 7   </xsd:sequence>
 8  </xsd:complexType>
 9
10  <xsd:complexType name="auto_t">
11   <xsd:sequence>
12    <xsd:element name="besitzer" type="besitzer_t"/>
13    <xsd:element name="modell" type="xsd:string"/>
14    <xsd:element name="bauart" type="bauart_t"/>
15   </xsd:sequence>
16   <xsd:attribute name="marke" type="xsd:string" use="required"/>
17   <xsd:attribute name="baujahr" type="xsd:integer" use="required"/>
18   <xsd:attribute name="info" type="xsd:string" use="optional"/>
19  </xsd:complexType>
20
21  <xsd:simpleType name="besitzer_t">
22   <xsd:restriction base="xsd:string">
23    <xsd:maxLength value="10"/>
24   </xsd:restriction>
25  </xsd:simpleType>
26
27  <xsd:simpleType name="bauart_t">
28   <xsd:restriction base="xsd:string">
29    <xsd:enumeration value="SUV"/>
30    <xsd:enumeration value="Kombi"/>
31    <xsd:enumeration value="Stufenheck"/>
32    <xsd:enumeration value="Cabrio"/>
33   </xsd:restriction>
34  </xsd:simpleType>
35
36  <xsd:element name="autos" type="autos_t">
37   <xsd:unique name="x">
38    <xsd:selector xpath="auto"/><xsd:field xpath="besitzer"/>
39   </xsd:unique>
40  </xsd:element>
41
42 </xsd:schema>
```

1a. Markieren Sie jene Stellen im folgenden XML Dokument, die in Bezug auf das gegebene XML Schema *ungültig* sind. Begründen Sie jeweils kurz, warum die Stelle ungültig ist. [13]

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <autos>
4    <auto marke="Ford" baujahr="2000">
5      <besitzer>M. Derntl</besitzer>
6      <modell>Focus Traveller 1.8i</modell>
7      <bauart>SUV Kombi</bauart>
8    </auto>
9
10   <auto marke="Hummer" jahr="2005">
11     <besitzer>A. Schwarzenegger</besitzer>
12     <bauart>SUV</bauart>
13     <modell/>
14   </auto>
15
16   <auto marke="Peugeot" baujahr="1996">
17     <besitzer>M. Derntl</besitzer>
18     <modell>306</modell>
19     <bauart>Stufenheck</bauart>
20   </auto>
21
22   <auto marke="Ferrari" baujahr="2008"/>
23
24   <auto marke="Aston Martin" info="Filmauto">
25     <besitzer>James Bond</besitzer>
26     <modell>DB 9</modell>
27     <bauart>Cabrio</bauart>
28     <bauart>SUV</bauart>
29     <radstand>3.5m</radstand>
30   </auto>
31
32 </autos>
```

1b. Erklären Sie, welche in dem XML Schema definierten Einschränkungen Sie mit einer DTD nicht definieren könnten (mit kurzer Begründung!) [8]

1c. Kreuzen Sie die *falschen* Aussagen an: [4]

- **O** Mit Attributgruppen kann man in XML Schema die Reihenfolge der Attribute eines XML Elements definieren.
- **O** Wird für ein Attribut kein Namespace Präfix angegeben, so befindet sich das Attribut im Default Namespace.
- **O** In einer WSDL Datei muss man immer ein XML Schema im <types> Abschnitt angeben.
- **O** In XSLT werden die im Stylesheet definierten Templates der Reihe nach auf das Quelldokument angewendet.
- **O** Der XPath Ausdruck `/*` kann mehrere Elemente als Ergebnis liefern.
- **O** Bei XML Elementnamen ist die Groß-/Kleinschreibung irrelevant.
- **O** Wohlgeformte XML Dokumente können niemals ungültigen Inhalt haben.

1d. Nehmen Sie an, es existiert ein XML Dokument, das gültig für das gegebene XML Schema ist. Erstellen Sie nun je einen XPath Ausdruck für folgende Abfragen:

i. Die Besitzer aller Autos der Marke “BMW”. [2]

ii. Die Baujahre aller Autos, deren Besitzer “Ali G.” heißt. [2]

iii. Das Modell des letzen in der Datei gespeicherten Autos. [1]

iv. Alle Baujahre in denen mindestens zwei Autos gebaut wurden. (Hinweis: dafür brauchen Sie XPath Achsen!) [3]

v. Die Anzahl der Autos der Bauart “Cabrio”, die vor dem Jahr 2000 gebaut wurden, und die nicht von der Marke “Mercedes” sind. [2]

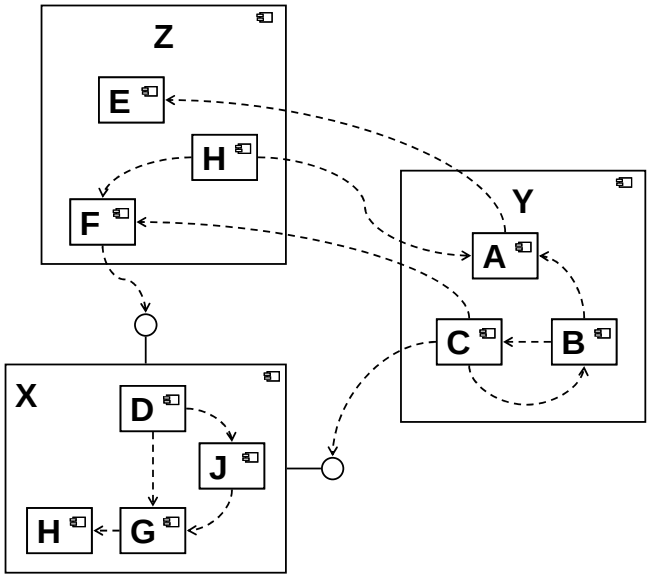
Aufgabe 2: Software-Architektur [35]

2a. Erklären Sie kurz, warum die Software-Architektur in der Evolution der Software-Entwicklung laufend an Bedeutung und Bewusstsein als Disziplin gewinnt und gewonnen hat. [4]

2b. Was versteht man allgemein unter "Mud-to-Structure" Patterns? Erklären Sie zwei dieser Patterns jeweils kurz in 2-3 Sätzen. [7]

2c. Welchen Zweck erfüllt der MVC Pattern? Aus welchen Komponenten besteht eine MVC Lösung, welche Aufgaben haben die Komponenten dabei und wie spielen die Komponenten zusammen? [10]

2d. Was versteht man in der Software-Architektur unter *Kopplung*, *Kohäsion*, und *Information Hiding*? Bewerten Sie die Einhaltung dieser Prinzipien durch die Komponenten im folgenden Architekturmodell (inkl. Begründung)? [14]



Aufgabe 3: Webanwendungen [20]

3a. Erklären Sie die Konzepte und Funktionsweise von *Cookies* und *Sessions* in dynamischen Webanwendungen. [10]

3b. Wofür benötigen Sie in einer Webservice-Definition (WSDL) die Abschnitte *Types*, *Messages*, *Port-Type* und *Binding*? Was wird in diesen Abschnitten definiert und wie hängen diese Abschnitte zusammen? [10]