«MATNR» «NACHNAME» «VORNAME»

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

Schriftliche Einzelprüfung zur Vorlesung des Wintersemesters 2008/09
Zweiter Prüfungstermin
Fr 20.03.2009

Prüfer: Dr. Michael Derntl

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 **Do 9.4.** im PISWI verfügbar sein.
- **Einsicht** ist **nach Terminvereinbarung** per E-Mail bis spätestens **Fr 24.4.**
- Zwei weitere Prüfungstermine werden im noch Sommersemester 2009 folgen.

Aufgabe 1: XML Technologien [38]

1a. Logische Struktur von XML. Beschreiben Sie so umfassend wie möglich folgende Bestandteile der logischen Struktur von XML. (Stellen Sie sich z.B. jeweils folgende Fragen: Was ist das? Wo befindet es sich in der XML Datei? Wie sieht es aus? Welche Bestandteile hat es bzw. kann es haben? Welche Ausprägungen kann es annehmen? etc.) [12]

- Wurzelelement:

- Element:

- Elementinhalt:

- Tag:

- XML Deklaration:

- Attribut:

1b. XSLT. Gegeben sind jeweils eine XML Datei und ein Stylesheet. Notieren Sie jeweils die Ausgabe bei Anwendung des Stylesheet auf die XML Datei. [14] = [2+4+8]

XML Datei:

```
<buch>
 <autor>C. Bukowski</autor>
 <titel>Hollywood</titel>
</buch>
```

XSL Stylesheet:

```
<xsl:template match="autor">
 <xsl:value-of select="text()"/>
</xsl:template>
```

Ausgabe:

XML Datei:

```
<buch>
 <autor>
   <titel>Dr.</titel>
   <vname>«VORNAME»</vname>
   <nname>«NACHNAME»</nname>
 </autor>
 <titel>Bring 'em on!</titel>
</buch>
```

XSL Stylesheet:

```
<xsl:template match="autor">
 Vorname: <xsl:value-of select="vname"/>
</xsl:template>
<xsl:template match="titel">
 Titel: <xsl:value-of select="."/>
</xsl:template>
<xsl:template match="nname">
 Nachname: <xsl:value-of select="nname"/>
</xsl:template>
```

Ausgabe:

XML Datei:

```
<A>
  <B a="47" b="«MATNR»">
   <C>x</C>
   <C t="woa">y</C>
   <C><D>muh</D><D/></C>
 </B>
 <B a="«NACHNAME»">
   <D>blah</D>
   <D k="1" r="3">dah</D>
 </B>
 <D>jabba</D>
</A>
```

XSL Stylesheet:

```
<xsl:template match="B[@*]">
  B: <xsl:for-each select="@*">
    <xsl:value-of select="."/> /
  </xsl:for-each>
  Mehr: <xsl:apply-templates select="*/*"/>
</xsl:template>

<xsl:template match="D">
  D = <xsl:value-of select="text()"/> !!!
</xsl:template>
```

Ausgabe:

«MATNR» «NACHNAME» «VORNAME»

1c. DTD. Gegeben ist folgende DTD:

```
<!ELEMENT m (k|x|z)>
<!ELEMENT k (s?,b)+>
<!ELEMENT x ((s|b)+|c*)>
<!ELEMENT z (#PCDATA)>
<!ELEMENT s EMPTY>
<!ELEMENT b EMPTY>
<!ELEMENT c EMPTY>
```

In der folgenden Tabelle steht jede Zeile für ein eigenes XML Dokument. Kreuzen Sie in jeder Zeile an, ob das XML *gültig* oder *ungültig* für die gegebene DTD ist! [12]

	gültig	**ungültig**
`<m><x><b/></x></m>`		
`<m><z>Sapper <lott/></z></m>`		
`<m><x><b/><s/><b/></x></m>`		
`<m><x><c/><c/><c/></x></m>`		
`<m><x><k/></x></m>`		
`<m><k><b/><b/></k></m>`		
`<m><z><s/><b/></z></m>`		
`<m><k>David Banner</k></m>`		
`<m><k><b/><s/></k></m>`		
`<m><x><s cool="night"/></x></m>`		
`<m><k/></m>`		
`<m><z>Hans</z></m>`		
`<m><k><b/><b/><s/></k></m>`		
`<m><x><s/><c/></x></m>`		

Aufgabe 2: Software Architektur [27]

2a. Architekturpatterns.

i. Was versteht man unter "Mud-to-Structure" Patterns und warum sind sie wichtig für komplexe Softwaresysteme? [3]

ii. Erklären Sie zwei "Mud-to-Structure" Patterns jeweils kurz. [6]

2b. Vergleichen Sie *Message Oriented Middleware* (MOM) und *Remote Procedure Call* (RPC) im Kontext eines beispielhaften von Ihnen gewählten Anfrage/Antwort-Ablaufs in einem verteilten System. (Anmerkung: beschreiben Sie den Ablauf verbal und erstellen Sie eine kleine Skizze) [6]

2c. Representational State Transfer (REST). Erläutern Sie Konzept, Eigenschaften und Unterschiede einer REST Architektur im Vergleich zu einer service-zentrierten Architektur anhand eines selbstgewählten konkreten Beispiels. [12]

Aufgabe 3: Webanwendungen [25]

3a. Webservices. Erklären Sie die jeweils ausführlich den Zweck und die Inhalte der folgenden Abschnitte einer WSDL Datei: [15]

i. Types:

ii. Messages:

iii. Port-Type:

iv. Binding:

v. Service:

3b. Webformulare:

i. Schreiben Sie ein HTML Formular, das ein Feld für die Eingabe einer maximal 5-stelligen Artikelnummer, ein Auswahlfeld für die Bewertung des Artikels (1 bis 5), sowie jeweils einen Knopf zum Abschicken und Zurücksetzen des Formulars beinhaltet. Definieren Sie, dass die Formulardaten mittels der Methode "POST" an das Skript "eintragen.php" übertragen werden sollen. (Anmerkung: nur der HTML Code für das Formular ist hier gefragt, nicht der Code der gesamten HTML Seite) [8]

ii. Was ist zwischen den Methoden GET und POST der Unterschied bei der Übertragung der Formulardaten an das verarbeitende Skript? [2]

«MATNR» «NACHNAME» «VORNAME»