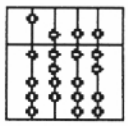


INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Semesterprüfung zur Vorlesung
"Softwarearchitektur verteilter Systeme"
WS 2005 / 06

Wo:	Technische Universität München Institut für Informatik Garching Raum FMI HS2
Wann:	Montag den 03.04.2006, 09:00 Uhr
Geprüfte Vorlesung:	Softwarearchitektur verteilter Systeme, WS 2005 / 06
Gelesen von:	Prof. Dr. Dr. h.c. Manfred Broy, Dr. Klaus Bergner, Dr. Marc Sihling
Bearbeitungszeitraum:	09:15 bis 10:30 (75 Minuten)
Erlaubte Hilfsmittel:	Keine

Aufgabe 1

1. (2 Punkte) Definieren und erläutern Sie den Begriff „Softwarearchitektur", wie er in der Vorlesung entwickelt wurde. *Lösung: Softwarearchitektur ist Menge von Komponenten und ihren Beziehungen untereinander. Gleichfalls ist Softwarearchiektur eine Teildisziplin des Software-Engineerings mit Definition von Rollen, Methoden, Techniken und Werkzeugen.*

2. (5 Punkte) Aus welchen Schichten besteht eine logische Drei-Schichtenarchitektur, welche Aufgaben haben die einzelnen Schichten und welche unterschiedlichen Möglichkeiten der Aufteilung einer Drei-Schichtenarchitektur auf Client und Server gibt es? Nennen sie drei Beispiele und skizzieren Sie sie grafisch.
Hier sind einige Varianten denkbar. Ad 1: Schichtenbeschreibung muss vorliegen für Präsentations-, Business- und Datenschicht (PBD). Dann kommen noch verschiedene Varianten der Verteilung der einzelnen Schichten dazu, z.B. P-B-D, PB-B-D, P-PB-D etc.

3. (3 Punkte) Welche Vorteile bietet der Einsatz einer dedizierten Steuerungsschicht? Nennen Sie drei Punkte!

   - *Einfache Konfigurierbarkeit der Prozesse ohne Programmierung*
   - *Erstellung der Prozesse mit graphischen Editoren*
   - *Möglichkeit zur Nutzung der definierten Prozesse durch unterschiedliche Präsentationen*
   - *Nutzung vorhandener Standards und kommerzieller Komponenten (z.B. WfEs) für GUI, Ablaufsteuerung etc.*
   - *Ggf. vereinfachte Einbindung / Anbindung von anderen Applikationen (Orchestrierung)*
   - *Ggf. vereinfachte Verteilung der Anwendung bzw. Kommunikation mit anderen Anwendungen*

4. (3 Punkte) Ein Architekt entscheidet sich dafür, bei der Entwicklung eines Systems modellbasiert vorzugehen. Aus einem anfangs erstellten, abstrakten und verständlichen Modell sollen dabei einmalig Programmrahmen für die Implementierung generiert und dann in der Folge manuell mit dem noch fehlenden Programmcode vervollständigt werden. Eine Rückübersetzung des Programmcodes in das Modell sowie eine Änderung bereits teilweise ausprogrammierter Programmrahmen erlaubt der verwendete Generator nicht. Nennen Sie mindestens drei Gründe, die für bzw. gegen diese Vorgehensweise sprechen!

   - *Das Modell erleichtert den Entwurf, da es eine abstrakte Sicht auf das System bietet.*
   - *Arbeitsersparnis für die initiale Erstellung der Programmrahmen.*

- *Änderungen, die die Programmrahmen und damit den Entwurf betreffen, sind schwierig:*
- *Entweder wird das Modell nicht nachgezogen und damit inkonsistent zur Implementierung (damit ist das Modell nicht als Dokumentation nutzbar).*
- *Oder es entsteht ein hoher Aufwand dafür, das Modell mit der Implementierung konsistent zu halten.*
- *Bei Änderungen der Anforderungen werden Modell und Code leicht inkonsistent, weil beide einheitlich nachgezogen werden müssten.*
- *Keine Möglichkeit zur Optimierung der Programmrahmen durch erneute Generierung.*

5. (2 Punkte) Vergleichen Sie Delegation und Vererbung hinsichtlich Ihrer Eignung zur Erweiterung von Komponentenframeworks.*Erweiterung per Vererbung erfordert ein Verständnis über die Interna des Frameworks, während sich Delegation der regulären Schnittstellen bedient und somit eine bessere Kapselung ermöglicht.*

6. (3 Punkte) Beschreiben Sie Idee und Zielsetzung des Konzepts „Architekturtreiber". Nennen Sie ein Beispiel.
Herstellung eines Zusammenhangs zwischen Anforderungen an die Architektur und prinzipiellen Wegen der Ausgestaltung. Damit die Möglichkeit, Auswirkungen abzuschätzen und die Architektur zu optimieren. Strukturierte Erfassung.

7. (3 Punkte) Erörtern Sie Vor- und Nachteile leichtgewichtiger Containerarchitekturen wie beispielsweise „Spring" im Vergleich zu schwergewichtigen Ausprägungen wie etwa „Enterprise Java Beans".*Hier ist viel denkbar. Z.B., einfache Container für einfache Anwendungen (Pojos auch außerhalb Container testbar), strenge Auflagen an EJBs hinsichtlich zu erfüllender Schnittstellen, etc.*

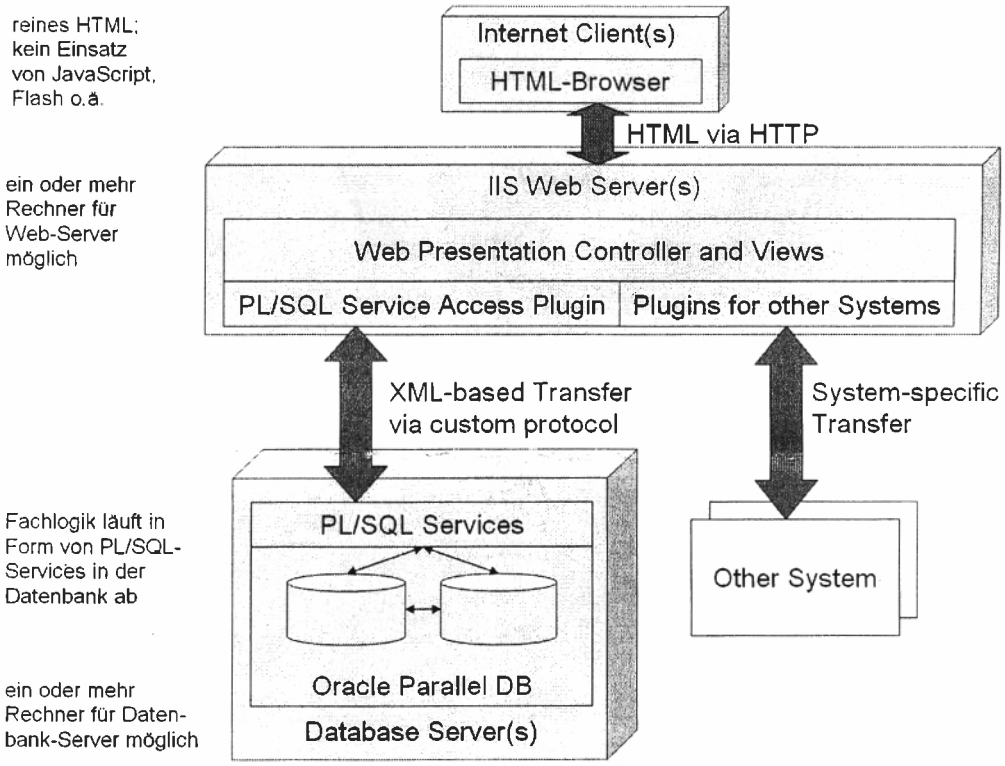
8. (2 Punkte) Nennen Sie zwei Mittel, bzw. Methoden zur Verhaltensbeschreibung von Komponenten.
Z.B.: MSCs, formale Spezifikationen, Zustandsautomaten...

Aufgabe 2

	Frage	Ja	Nein
2.	Threads sind leichtgewichtiger als Prozesse.*(Wahr)*		
3.	Round Robin Architekturen für eingebettete Systeme benötigen kein Echtzeitbetriebssystem.*(Wahr)*		
4.	Platform-Independent Model und Platform-Specific Model nach MDA sind unter Verwendung der UML im vierschichtigen Meta-Modell beide derselben (Meta-)Ebene zugeordnet.*(Wahr – beide liegen (z.B. in Form von Klassendiagrammen) auf M1)*		
Welche der folgenden Aussagen treffen für eingebettete Systeme mit harten Echtzeitanforderungen zu?			
5.	Sie fordern grundsätzlich eine höhere Leistung von der Basishardware als betriebliche Informationssysteme.*(Falsch – Sobald die geforderte Reaktionszeit erfüllt ist, bringt schnellere Basishardware keinen weiteren Nutzen.)*		
7.	Sie fordern grundsätzlich eine niedrigere Leistung von der Basishardware als betriebliche Informationssysteme.*(Falsch – Hängt von der Anwendung ab.)*		
8	Solange die geforderten Reaktionszeiten erfüllt werden können, ist der Einsatz „schnellerer" Basishardware nicht erforderlich.*(Wahr)*		
9.	Ersetzt man die vorhandene Basishardware durch schnellere, kann das optimierte System harte Echtzeitanforderungen besser erfüllen.*(Falsch – wenn schon das nicht optimierte System die Anforderungen erfüllt, bringen Optimierungen nichts.)*		
In einem transaktionsbasierten, workflow-gesteuerten System soll aus Kostengründen die proprietäre Workflow-Engine SimpleFlow des Herstellers Simpelheimer & Co. eingesetzt werden. Der Architekt soll allerdings dafür Vorsorge treffen, dass sie in Zukunft gegebenenfalls leicht durch die nicht dazu kompatible Workflow-Engine DeluxeWork des Herstellers SuperComponents Inc. ersetzt werden kann. Welche der folgenden Strategien sind dazu sinnvoll?			
10.	Entwurf eines Adapters, der die Workflow-Engine kapselt.*(Wahr)*		
11.	Einbindung der Workflow-Engine als Transaktions-Koordinator statt als transaktionale Ressource und damit Umkehrung der Komponenten-Abhängigkeiten.*(Falsch – unsinnig)*		
12.	Verwendung einer objektorientierten Datenbank zur Ablage der Workflow-Daten.*(Falsch – kann ggf. sogar neue Kompatibilitätsprobleme schaffen)*		
13.	Automatisierte Generierung von Web-Services für die Schnittstellen von SimpleFlow.*(Falsch – Umstieg auf eine andere Schnittstellen-Technologie löst das grundlegende Migrationsproblem nicht)*		
14.	Eine mehrschichtige Anwendungsarchitektur erzielt immer dann die beste Performance, wenn jede logische Schicht auf einem eigenen (physischen) Netzwerkknoten lokalisiert wird.*(In dieser Form falsch, wegen Netzwerklaufzeit etc.)*		

Aufgabe 3: Eigenschaften einer Software Architektur

Software-Architekt Max Richtig erhält die Aufgabe, die Software-Architektur für das Basis-Produkt eines eCommerce-Startups zu entwickeln. Das System soll über das Internet für viele tausend Benutzer verfügbar sein und ihnen im Wesentlichen die Möglichkeit bieten, einfach strukturierte Daten formularbasiert einzugeben und abzurufen. Die Software-Architektur soll mindestens für die nächsten zehn bis fünfzehn Jahre als Basis einer ganzen Produktlinie tragfähig sein. Max Richtig erarbeitet eine Software-Architektur und dokumentiert sie in Form eines informellen Boxes-and-Line-Diagramms.



Anmerkungen:

- PL/SQL ist eine Oracle-spezifische, prozedurale Programmiersprache, die besondere Konzepte zur Einbindung von SQL-Queries hat. PL/SQL-Programme laufen direkt auf dem relationalen Datenbanksystem ab.
- Machen Sie zur Beantwortung der Fragen geeignete Annahmen und stellen Sie diese plausibel dar.

Fragen

1) (3 Punkte) Identifizieren Sie die logischen Schichten der Architektur!
   - *Präsentationsschicht, kombinierte Anwendungs- und Datenhaltungsschicht in Form von PL/SQL-Code mit eingebundenen SQL-Queries (ggf. auch zur Not noch motivierbar: Anwendungsschicht in Form von PL/SQL-Prozeduren, Datenhaltungsschicht in Form von SQL-Queries)*
2) (3 Punkte) Identifizieren Sie die verteilten physischen Schichten der Architektur!

- *Präsentationsschicht (Browser), Präsentationsschicht (Web Application Server), Anwendungs- und Datenhaltungsschicht (Datenbank-Server)*

3) (2 Punkte) Bewerten Sie die Architektur hinsichtlich Skalierbarkeit, Performanz und Ausfallsicherheit!
   - *Skalierbarkeit insgesamt sehr gut, weil: Skalierung Datenhaltungsschicht abhängig von verteiltem DB-System – beim Einsatz einer modernen verteilbaren DB sehr gut; Präsentationsschicht durch Hinzufügen weiterer Webserver sehr gut skalierbar*
   - *Performance insgesamt gut, weil: Datenhaltungsschicht optimiert in PL/SQL direkt auf DB, Präsentationsschicht einfach aufgebaut und performant – negativ: keine oo Zugriffsschicht und kein Objektcache – jede Anfrage benötigt Roundtrip zur DB, Performance einer reinen HTML-GUI schlecht - jede Aktion braucht Roundtrip zum Server*
   - *Ausfallsicherheit insgesamt sehr gut, weil: Komponenten jeder Schicht können redundant gehalten werden (verteilte DB-Server, mehrere Webserver)*
4) (1 Punkt) Welche potenziellen Nachteile hat die Architektur?
   - *Programmierung der Anwendungslogik in PL/SQL: Keine Datenunabhängigkeit, Abhängigkeit von Hersteller Oracle, umständliche Programmierung von Anwendungslogik und Services, da keine objektorientierte Anwendungs-schnittstelle (sondern direkter Zugriff auf Tabellen), kein Zugriff auf Drittsysteme (Other Systems) via PL/SQL vorgesehen (also wieder: keine einheitliche Anwendungsschnittstelle)*
   - *Navigationsbeziehungen in Web Presentation Controller abgelegt – scheinen nicht einfach für andere GUIs verwendbar wenn später Multi-Channel-Architektur erforderlich wird*
5) (1 Punkt) Ist die Software-Architektur kurzfristig geeignet, die nötige Funktionalität zu realisieren? Ist sie längerfristig die richtige Wahl?
   - *Kurzfristig erfüllt sie die Anforderungen (performant, geeignet als Basis für Funktionalität). Längerfristig ist sie unflexibel, weil Fachlichkeit und Technik nicht ausreichend getrennt sind (Abhängigkeit von DB, fachliche Funktionalität liegt nicht technikunabhängig vor) – positiv dabei ist allerdings, dass sie eine Service-Schnittstelle hat und deshalb leicht integriert werden kann*