Klausur zur Vorlesung „Verteilte Systeme“ im SS 2007 Prof. Dr. Odej Kao

24. Juli 2007

Name:	
Vorname:	
Matrikelnummer:	
Studiengang:	
E-Mail:	

Schreiben Sie zunächst sofort Ihren Namen und Matrikelnummer auf die Klausur!

- *Lassen Sie die Klausur zusammengeheftet!*
- *Die Klausur dauert 75 Minuten und umfasst 6 Aufgaben auf 14 Seiten.*
- *Die Klausur ist bestanden, wenn mindestens 50% der Punkte erreicht wurden.*
- *Es sind keine Hilfsmittel zugelassen (insbesondere keine Taschenrechner und Handys)!*
- *Abschreiben und abschreiben lassen führt zum Nichtbestehen der Klausur!*
- *Benutzen Sie kein eigenes Konzeptpapier bzw. Schmierpapier. Sie bekommen bei Bedarf Papier von der Klausuraufsicht!*
- *Wenn Sie Konzeptpapier abgeben wollen, dann nur mit Namen und Matrikelnummer!*
- *Kennzeichnen Sie Ihre Lösung eindeutig. Es wird keine Lösung gewertet, wenn Sie zu einer Aufgabe mehr als eine Lösung abgeben.*
- *Bleistift, rote Stifte und Tipp-Ex sind nicht zugelassen!*

☐ *Ja, ich möchte mein Ergebnis per Email erhalten*

Aufgabe	**1**	**2**	**3**	**4**	**5**	**6**	**Σ (74)**
Punkte	**14**	**12**	**14**	**14**	**10**	**10**	
Erreicht							

Note

Aufgabe 1: Grundlegendes [6+4+4=14 Punkte]

a) Ein Client reagiert auf eine verlorene Ergebnisnachricht eines zustandsverändernden Servers durch ein wiederholtes Senden der Request-Nachricht. Der zuständige Server befolgt die At-most-once Semantik. Erläutern Sie die Vorgehensweise des Servers auf die wiederholte Nachricht. Skizzieren Sie dazu die notwendige Datenstruktur und den Serveralgorithmus zur Bearbeitung der Anfrage.

b) Skizzieren Sie die Architektur einer Rechnerfarm für Videodownloads. Die Videos werden in drei verschiedene Qualitätsstufen angeboten. Die Auswahl wird automatisch getroffen, nachdem der Benutzer die Bandbreite seiner Netzwerkverbindung angegeben hat. Für jede Videokategorie stehen wiederum je drei gespiegelte Server zur Verfügung. Sorgen Sie dafür, dass die Last gleichmäßig verteilt wird.

c) Welche Funktionalität besitzt ein Beobachterobjekt, wenn dieses als Benachrichtigungsfilter ausgelegt wird? Geben Sie ein konkretes Beispiel für den Einsatz eines solchen Filters.

Aufgabe 2: Web Services und Middleware [4+4+4=12 Punkte]

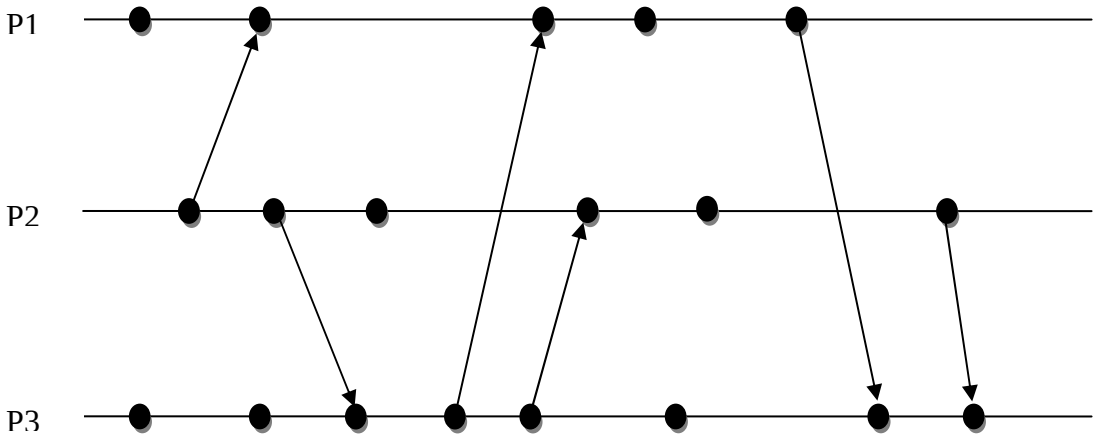
a) Geben Sie stichwortartig an, wozu eine WSDL-Datei benötigt wird und welche Bereiche darin enthalten sind.

b) Beschreiben Sie die Aufgabe des Object Request Brokers (ORB) in CORBA und erläutern Sie das Zusammenspiel mit dem Skeleton.

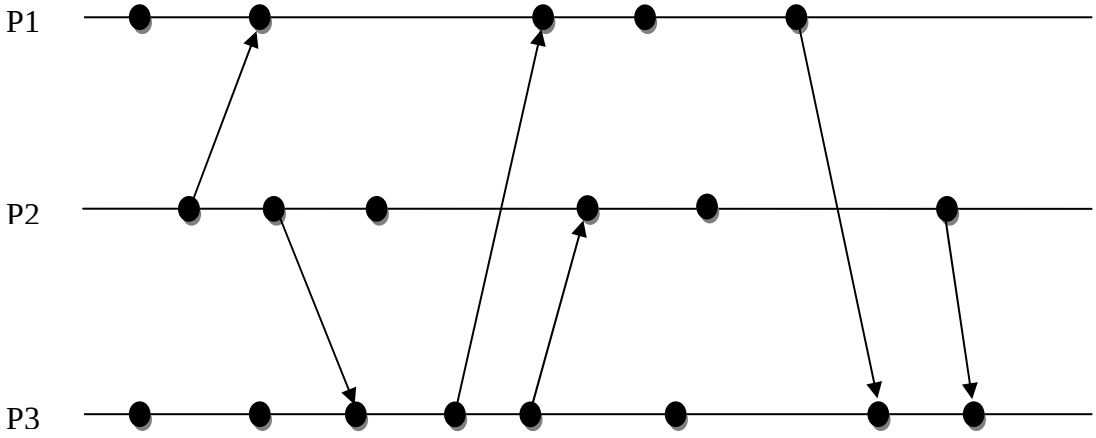
c) Beschreiben Sie die Schritte für das Deployment und für die Ausführung eines in C# entwickelten .NET Programms.

Aufgabe 3: Algorithmen [4+6+4=14 Punkte]

a) Geben Sie die Vektorzeit für jedes Ereignis im folgenden Beispiel an.

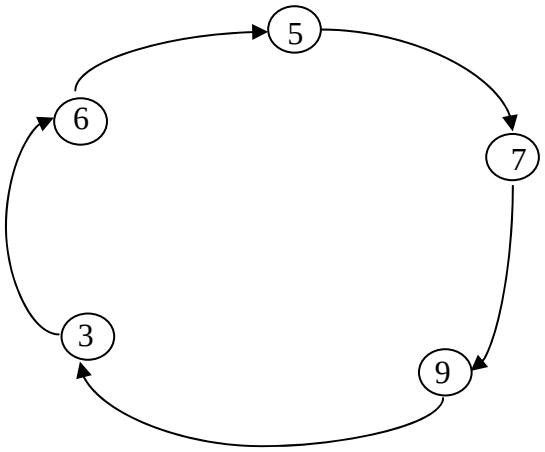


Ersatzvorlage



b) Erläutern Sie den Algorithmus von Maekawa und zeigen Sie ein Beispiel, in dem eine Deadlock-Situation entstehen kann.

c) Es sei folgender Ring für einen Wahlalgorithmus gegeben. Die Nummern in den Knoten stellen die ID dar. Der Knoten mit der ID 5 initiiert die Wahl (höchste ID gewinnt). Geben Sie die Aktionen an, die von jedem Knoten im Ring ausgeführt werden.



Aufgabe 4: Dateisysteme und Sicherheit [4+4+6=14 Punkte]

a) Beschreiben Sie das Callback Konzept von AFS (Funktionalität und Einsatz).

b) Wie wird Tunneling mit IPsec realisiert?

c) Beschreiben Sie den zweiten Kerberos-Interaktionsschritt zwischen Client und TGS:
   - Welche Eingaben werden vom Client gesendet?
   - Welche Aktivitäten werden vom TGS durchgeführt?
   - Welche Daten werden zum Client zurückgesendet?
   - Wie verarbeitet der Client die ankommenden Daten?
   - Welche Elemente werden zum beantragten Server gesendet?

   Verwenden Sie die Notation aus der Vorlesung.

Aufgabe 5: Verteilte Transaktionen und Replikation [6+4=10 Punkte]

a) Skizzieren Sie 3PC. Wie endet das Terminierungsprotokoll, wenn sich mindestens ein Teilnehmer nach dem Fehler im Zustand preCommited und kein Teilnehmer im Zustand commited ist. Erläutern Sie Ihre Antwort.

b) Beschreiben Sie das Voting Verfahren für Replikation und erläutern Sie die grundlegenden Regeln für die Stimmenverteilung auf die Knoten.

Aufgabe 6: Realisierung mit AXIS, Sockets, .NET [4+6=10 Punkte]

a) Erläutern Sie das Instant-Deployment von AXIS zum Deployen eines Java Webservices. Welche Schritte muss der Entwickler tun und worin unterscheidet sich der erste vom zweiten Serviceaufruf?

b) Die folgenden beiden Javaprogramme bilden zusammen eine PingPong-Anwendung auf Basis der Socketprogrammierung. Es fehlen an den gekennzeichneten Stellen socket-relevante Ausdrücke. Ergänzen Sie die nötigen Ausdrücke in ihrer korrekten Notation.

PingPongServer.java

```
import java.net.*;
import java.io.*;

public class PingPongServer {
  public static void main(String[] args) {
    try {

        [          ] MyServerSocket = new [                    ];

        [          ] MyClientSocket = MyServerSocket.[          ]();

        PrintWriter out =
            new PrintWriter([                                   ]());

        BufferedReader in =
            new BufferedReader(
            new InputStreamReader([                             ]()));

        //auf ping warten und mit pong antworten
        String inputline = null;
        while ((inputline = in.readLine()) != null) {
          System.out.println("Client: " + inputline);
          if (inputline.equals("ping")) {

              out.[          ]("pong");

              out.flush();
              System.out.println("Server: pong");
          }
          in.close();
          out.close();
          MyClientSocket.close();
        }
    } catch (Exception e) {
      System.err.println(e.getMessage());
    }
  }
}
```

PingPongClient.java

```
import java.net.*;
import java.io.*;

public class PingPongClient {
  public static void main(String[] args) {
    try {

      [                ] MySocket =

            new [                ]([                                ]);

      PrintWriter out =
            new PrintWriter([                                ]());

      BufferedReader in =
            new BufferedReader(
            new InputStreamReader([                                ]());

      BufferedReader stdIn =
            new BufferedReader(new InputStreamReader(System.in));

      //Eingabe erwarten und senden ...
      String command;

      out.[                ](command = stdIn.readLine());

      out.flush();
      System.out.println("Client: " + command);

      //Antwort empfangen
      String fromServer;
      while ((fromServer = in.readLine()) != null) {
        System.out.println("Server: " + fromServer);
      }
    } catch (Exception e) {
      System.err.println(e.getMessage());
    }
  }
}
```