Fragenkatalog
Verteilte Systeme VO
ICT Palensky

Fragen bis einschließlich 15.05.2007

noch keinem Kapitel zugeordnet:

1. **Ein Betriebssystem A läuft ununterbrochen, fällt aber jede Minute eine Millisekunde aus. Ein Betriebssystem B fällt niemals im Betrieb aus, muss aber jeden Tag 1 Stunde lang außer Betrieb genommen und gewartet werden.**
   F1: Welches System ist am Tag mehr verfügbarer?
   F2: Welches System ist (während des Betriebs) zuverlässiger? Seitenangabe
   - Das Betriebssystem A ist verfügbarer, da eine Millisekunde Ausfallzeit bei den meisten Arbeiten kaum auffällt.
   - Das Betriebssystem B ist zuverlässiger, da es niemals im Betrieb ausfällt. Außerdem müsste der Zeitpunkt der Wartung bekannt sein und optimal gewählt sein.

2. **Welche zwei Möglichkeiten gibt es Synchronisierungsinformationen auszutauschen Seitenangabe**

3. **3 Mögliche Probleme bei Time Sync Seitenangabe**

   IEEE1588 Standard: master failure results in a new “master election”
   Single point of failure: Master
   - Master fault causes a new master election
   - Babbling idiot problem

   Master election
   - Up to 10 sync periods

   Efficiency master-slave vs. democratic (#of links)

4. **Anforderungen an Sicherheitsarchitektur Seitenangabe**

5. **graphik mit dem routing? Seitenangabe**

was für werte kommen da rein? ich bin mir nicht ganz sicher ob das stimmt, aber ins erste kastl die adressen von sender und empfänger und ins zweite dann was vom router + den empfänger...

Kapitel 1: Grundlagen

6. Nenne 4 Eigenschaften von zentralen Systemen!

Siehe Frage 7

7. Nenne 2 Vorteile von Verteilten Systemen sowie 2 von zentralisierten Systemen! S3

Verteilte Systeme	zentralisierte Systeme
Vorteile: • einfacher zu modellieren/ programmieren • besser zu skalieren • kein „single point of failure“ = einzelne Fehlerstelle die Komplettausfall des Systems nach sich ziehen (fehlertolerante Systeme)	Vorteile: • einfache Handhabung (eine Ort für Konfiguration, Wartung, ...) • günstigere Ressourcen • konsistente Zustände („keine“ Kommunikation innerhalb)
Nachteile: • komplexe Nebeneffekte (state machine with permanent „race conditions“?) • komplexes Management (Netzwerk, Konfiguration, etc.) • teurere Ressourcen (n-mal Netzanschluss. Speicher, Gehäuse, …)	Nachteile: • Single point of failure • Schlechter zu skalieren • Komplexe Algrithmen/Modelle

8. Unterschiede zwischen NOS und DOS (6/19) S6ff

	System Images	Autonomie der Knoten (nodes)	Fehlertoleranz
NOS	n	Höher	niedrig
DOS	1	niedriger	hoch

Network Operating System:

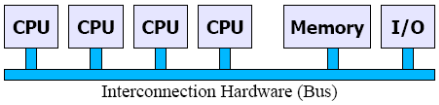
- der user erkennt n Maschinen
- explizit (Datentransfer, remote login, etc.)
- n Kernel laufen (ev. identische)
- Knoten sind verantwortlich für sich selbst und die Netzwerkdienste

Distributed Operating System:

- transparent
- einzelnes System Image
- organisiert und verteilt Aufgaben, Ressourcen, ….
- dynamische Bereitstellung der Ressourcen
- 1 Kernel
- Fehlertoleranz
- „true“ DOS

9. Nennen Sie 4 Eigenschaften/Charakteristika von Parallel Computing Systems S9

- CPUs teilen sich
  - memory ressources
  - bus
  - clocking
  - etc



- ist ein dicht gekoppeltes System:
  - gemeinsame Uhr (gemeinsame Zeit, keine Synchronisationsschwierigkeiten)
  - Hohe Bandbreitenkommunikation
    - zwischen Systemteilen
    - zwischen Systemteilen und Memory
  - physisch dicht (1 Gehäuse, 1 Motherboard, 1 Rückwandplatine, etc.)
  - logisch dicht (parallele Algorithmen, gemeinsamer Sourcecode)

- hat gute Performance/Durchsatz

10. Bustopologie S13-15

Eine **Bus-Topologie** (Linien- oder Strangtopologie) besteht aus einem Hauptkabel, dem Bus, an das alle Geräte und zwei Endwiderstände angeschlossen sind. Diese Abschlusswiderstände mit dem Leitungswellenwiderstand (Z = 50 Ohm bei Koaxialkabel) dienen zu Verhinderung von Reflexionen. Der Anschluss zwischen den Geräten (also Netzkarten) und Hauptkabel erfolgt über T-Stücke.

Zugriffsverfahren (z.B. CSMA/CD) versuchen zu verhindern, dass sich die Teilnehmer gegenseitig stören. Sie regeln, welcher Teilnehmer die gemeinsame Leitung – den Bus – zu welchem Zeitpunkt zur Verfügung hat.

Bei diesem Verfahren treten folgende Probleme auf:

- Während des Datenverkehrs muss jeder Teilnehmer jede Sendung mithören. Dadurch steigt die Belastung (Strom) der Ausgangsbaugruppen des Senders mit der Anzahl der Teilnehmer am Bus.
- Feldbussysteme können sich über einen Bereich von mehreren hundert Metern erstrecken. Hier ist die Leitungslänge im Vergleich mit der Wellenlänge der Übertragung nicht mehr vernachlässigbar klein. Um störende Reflexionen zu vermeiden, werden Busabschlusswiderstände benötigt, die die Ausgänge des Senders ebenfalls mit höheren Strömen belasten. Kleinere Feldbussysteme können dennoch sehr gut nach dem Bus-Prinzip vernetzt werden.

Vorteile:

- Der Ausfall eines Rechners hat keine Konsequenzen
- Nur geringe Kosten, da nur geringe Kabelmengen erforderlich sind
- Einfache Verkabelung und Netzerweiterung
- Es werden keine weiteren Rechner zur Übermittlung der Daten benötigt

Nachteile:

- Alle Daten werden über ein einziges Kabel übertragen
- Datenübertragungen können leicht abgehört werden
- Eine Störung des Übertragungsmediums an einer einzigen Stelle im Bus (defektes Kabel, lockere Steckverbindung, defekte Netzwerkkarte) blockiert den gesamten Netzstrang (die Suche nach der Fehlerquelle ist dann oft sehr aufwändig)
- Es kann immer nur eine Station Daten senden. Während der Sendung sind alle anderen blockiert (Datenstau)

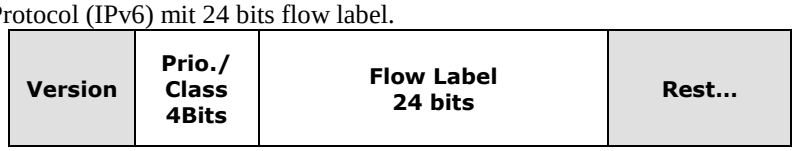
11. Was ist QoS? S16

Liefert Daten auf verlässliche Weise.
Optimum: Datenverlust, zeitverzögert, Verzögerungscharakteristik, Effizienz

- Latenzzeit: die Verzögerung der Ende-zu-Ende-Übertragung
- Jitter: die Abweichung der Latenzzeit von ihrem Mittelwert
- Paketverlustrate: die Wahrscheinlichkeit, dass einzelne IP-Pakete bei der Übertragung verloren gehen
- Durchsatz: die pro Zeiteinheit im Mittel übertragene Datenmenge

12. Was ist ein Flow bei Quality of Service (QoS)? S22

Flow: Sequenz von verwandten Paketen von der Quelle zum Ziel
-> Internet Protocol (IPv6) mit 24 bits flow label.



13. What is real time? S23

- An aspect of time can be guaranteed
  e.g. Latency, response time, execution time, etc.

- IT IS NOT Being fast
- Example: Heating Controller (Execution time of 30sec is OK)

14. Welcher Server ist stabil, stateful oder stateless bzw. welcher ist stabiler: stateful oder stateless? S29,30 und 65,66

- Stateful: Server Maintains the client's state from one RPC to the next
- Stateless: RPCs are unrelated / All information must come with call / Easier, simpler
- Stabil: Individual Agents should be stable

Meiner Meinung nach ist der Stateless Server stabil da jede Anfrage als eigene Transaktion betrachtet wird. Außerdem muss er keinen Speicher vorreservieren (macht der stateful server schon falls der client abschmiert).

Die Frage ist nur was als stabil gemeint ist - die Verbindung zum stateful server ist stabiler, dafür ist der stateless server in seinem Aufbau einfacher gehalten...

da der stateless fehlertoleranter ist (frag mich jetzt nicht auf welcher seite das stand) ist er meiner meinung nach stabiler ...

seh ich auch so, vor allem weil ich beim stateless server auch kein speicherproblem hab und die DoS-Attacke mit anderen Mitteln verhindern könnte... oder etwa nicht? ich mein dann müsst ja jeder apache in die knie gehn weils so viele leute lustig finden DoS-Attacken durchs netz an die armen kleinen indianer zu schicken...

beispiel für einen stateful server wäre übrigens Apache Tomcat. mah ich liebe Wikipedia wenn das was drin steht einigermaßen sinn macht.

stateless, weil "survive failures" ... schätz ich

15. Wie kommt es zu einem Deadlock? S31ff

Mutual exclusion
Hold and wait
No pre-emption
Circular wait

16. Beschreiben sie 3 Arten zur Vorbeugung eines Deadlocks! S33

Collective Request
- Request whenever, no hold and wait!

Ordered Request
- Total ordering of resources

Vorrecht
- Resourcen sind zugeordnet

17. Messagetransfer von Client zu Server (Was passiert, wenn request nicht ankommt?) S34

Message gets lost
Response gets lost
Recipient (Server) crashes

18. Was passiert wenn ein immutable file geschrieben wird? S37

- neue Idee. Nicht wirklich veränderbar .... (Hä????)
- CFS Cedar File System
- Dateien werden nie verändert, sobald sie angelegt werden
- Update Mechanismus
  - test.txt!1 gespeichert, vom Client gelesen
  - Client verändert Inhalte
  - speichert test.txt!2
- Performanceerhöhung

- nur Veränderungen speichern
- (sehr) alte Versionen löschen

Kapitel 2: Middleware

19. Nenne 4 Arten von Transparenz (oder: Anforderungen an) bei Middleware. S50

- Scaling Transperancy
- Performance Transperancy
- Failure Transperancy
- Migration Transperancy
- Replication Transperancy – verbirgt die Existenz und den Ort von Kopien
- Concurrency Transperancy
- Access Transparency – verbirgt Unterschiede in der Datendarstellung und wie der Zugridd auf eine Ressource erfolgt
- Location Transperancy – verbirgt, wo sich eine Reesource befindet
- Structure Transperancy – mehrere File Server erkennt man als einen File Server+
- Naming Transperancy – Ort sollte nicht wichtig sein

20. Welche Schwierigkeiten gibt es bei verteilter Garbage Collection? S55

- Ansätze:
  - Managed by the underlying plattform (e.g. distributed OS)
  - the broker identifies the number of live remote references (active connections count)
  - the server counts the number of references and the clients need to cooperate
- Distributed reference counting: es ist schwierig die zyklischen Referenzen zu identifizieren

21. Garbage Collector: Vor- und Nachteile. S55?

Problem
> Objektreferenzen über Rechnergrenzen hinweg
> Verständigung mit asynchronen, evtl. unzuverlässigen Nachrichten
> Speicherverwaltung ist komplex – sollte nicht dem Benutzer zugemutet werden

Motivation
> unbenötigter Speicherplatz soll frühzeitig freigegeben werden
> Speicherverwaltung ist komplex – sollte nicht dem Benutzer zugemutet werden
> Transparenz

22. Was ist persistence Referenz (11/19) S56

- die Lebenszyklen von Servern und Clients sind nicht synchron
- automatische Reaktivierung von einem schlafenden Server
- Server-Objekte speichern ihren Status vor dem Timeout
- träges Laden im Aktivierungsprozess

23. Was ist ein Stub/Skeleton proxy S58

- der Client ruft eine remote method auf, der Aufruf wird zuerst zum Stub weitergeleitet
- der Stub ist für das Senden des remote calls zum server-side skeleton verantwortlich
- der Stub öffnet ein Socket zum Remote Server, führt (marshalling – wandelt Datenelemente in ein Format um, um sie in einer Nachricht an den Empfänger (Skeleton) zu schicken) die Objekt Parameter zum Skeleton und leitet den Datenstrom (data stream) an diesen weiter
- ein Skeleton beinhaltet eine Methode welche die Remote-calls empfängt, unmarshals – wandelt die Nachricht wieder so um, das die gleich Datenstruktur wieder hergestellt werden kann - die Parameter und ruft die eigentliche Remote Object Implemenation auf. Die Returnwerte werden wieder an den Stub zurück geschickt.

24. Erklären Sie die RMI Architektur. S58

- Der Server muss zuerst sein Name mit der Registry verbinden (bzw. einbinden)

- der Client schlägt in der Registry den Servernamen nach um Remote Referenzen aufzubauen
- der Client ruft eine remote method auf, der Aufruf wird zuerst zum Stub weitergeleitet
- der Stub ist für das Senden des remote calls zum server-side skeleton verantwortlich
- der Stub öffnet ein Socket zum Remote Server, führt (marshalling – wandelt Datenelemente in ein Format um, um sie in einer Nachricht an den Empfänger (Skeleton) zu schicken) die Objekt Parameter zum Skeleton und leitet den Datenstrom (data stream) an diesen weiter
- ein Skeleton beinhaltet eine Methode welche die Remote-calls empfängt, unmarshals – wandelt die Nachricht wieder so um, das die gleich Datenstruktur wieder hergestellt werden kann - die Parameter und ruft die eigentliche Remote Object Implemenation auf. Die Returnwerte werden wieder an den Stub zurück geschickt.

25. Was ist ein Skeleton / Server-side Proxy? S58

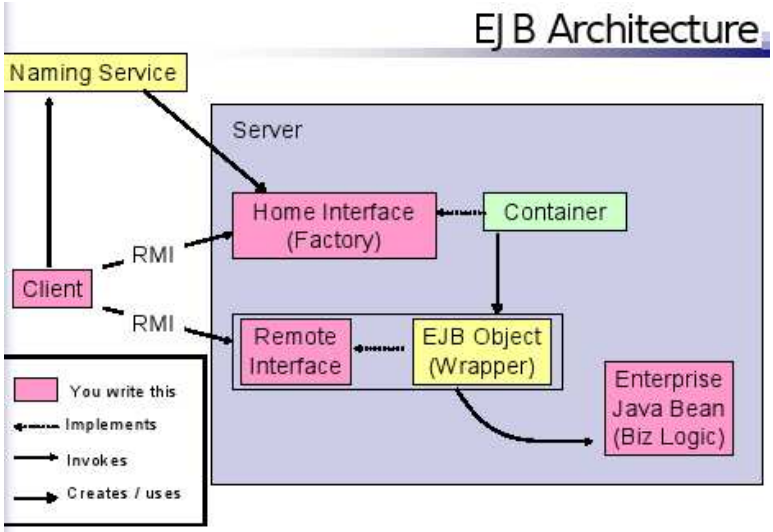
siehe Frage oben.

26. Was ist ein Stub? S58

siehe Frage oben.

27. Erklären Sie die EJB Architektur. S62

- An Enterprise Java Bean (EJB) is a component that provides reusable business logic functionality and/or a representation of a persistent business entity
  - Enterprise Java Beans (EJB) sind standardisierte Komponenten innerhalb eines J2EE-Servers (Java 2 Enterprise Edition). Sie vereinfachen die Entwicklung komplexer mehrschichtiger verteilter Softwaresysteme mittels Java.
- An EJB Container executes an EJB due to a client request.
  - Provides the plumbing necessary to execute the EJBincluding
  - Client uses an interface to access the Bean indirectly
  - EJB-Container definieren zusätzliche containerspezifische Eigenschaften. Zum Zeitpunkt der Installation können diese Eigenschaften je nach Container auf unterschiedliche Art angegeben werden – in Java-Properties-Dateien, XML-Dateien oder interaktiv.
- A deployment descriptor (Konfiguration)describes the structure of the Bean and how to execute the Bean as part of an application
  - Dieser Deployment Descriptor ist eine XML-Datei, in der Eigenschaften von EJBs definiert werden, die nicht hart codiert sind. Dazu zählen:



28. Welche Arten von Persistentmanagement gibt es S66

- Bean managed: (programmatisch)
  - Bean provider schreibt Routinen um einen Zugang zum Data Store herzustellen
    - deklariert Instance Variablen, die die persistenten Daten beinhalten
    - Finder method implementation wird vom Bean provider geschrieben
  - Container ruft diese Routinen zu einem angemessenen Zeitpunkt im Lifecycle auf
    - ist gebräuchlicher, when der zu Grunde liegende Speicher (store) eine Applikation ist.
- Container managed: (deklarativ)
  - Erlaubt Bean logisch unabhängig von der Datenquelle zu sein

        - z.b. gleicher Code für relatiionale Datenbank, IMS Datenbank, etc.
    - Container erzeugt Code um den Datenspeicher (data store) zu erreichen
        - Deployer maps the fields of the Bean to the columns of the data base
            - Bean provider describes Bean's fields and relationships to other Bean in development descriptor
        - Container may use lazy access methods and caching
        - Finder methods are described in the deployment descriptor
            - Description is in EJB QL
            - Implementation is generated when Bean is deployed
    - Virtual fields are used in the Bean to contain the persitant data
        - Access is via getXXX/setXXX methods.

29. Wie sieht die generelle Architektur eines Verzeichnisdienstes aus? S68

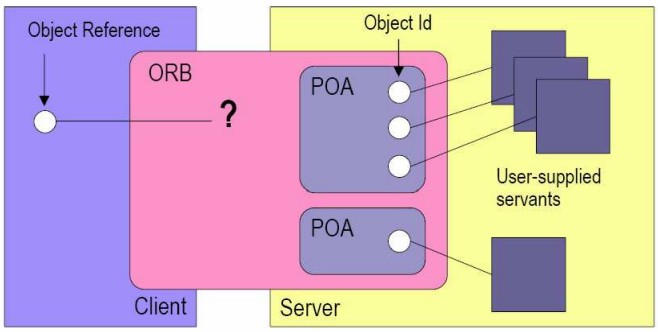
Ein Verzeichnisdienst (englisch: Directory Service (DS)) ist eine im Netzwerk **verteilte hierarchische Datenbank** (es bildet die Realwelt durch eine hierarchische Baumstruktur ab. Jeder Record hat also genau einen Vorgänger und somit genau ein Satz an der Wurzel der so entstehenden Baumstruktur), die auf dem Client-Server Prinzip basiert. In dieser Datenbank können beliebige Informationen gespeichert werden.
Die Einträge in der Datenbank können verglichen, gesucht, erstellt, modifiziert und gelöscht werden.
Meistens wird lesend auf die Daten eines Verzeichnisdienstes zugegriffen. Veränderungen an den Einträgen dieser Datenbank sind sehr selten. Aus diesem Grund bieten Verzeichnisdienste lesend eine wesentlich geringere Zugriffszeit als andere Datenbanken.
Der Aufbau der Directory Services erfolgt prinzipiell nach dem sogenannten X.500-Standard.

30. Portable Object Adapter (POA) S77

31. Welche Aufgaben hat der Portable Object Adapter in CORBA? S77

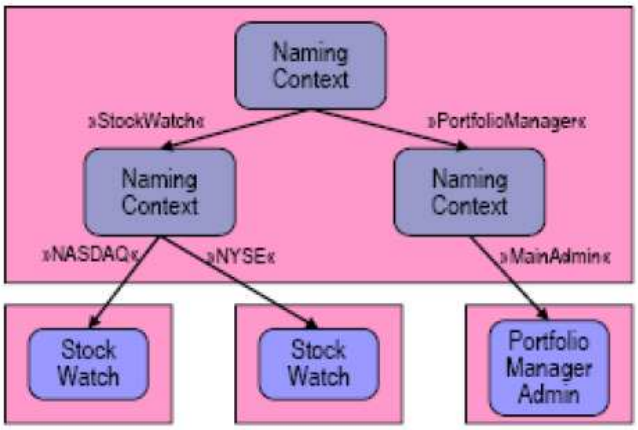
- wurde in CORBA 2.2 vorgestellt
- ersetzt under-specified BOAs (Basic Object Adapter)
- Allow construction of object implementations that are portable between different ORB products
- Provide support for objects with persistent identities
- Provide support for transparent activation of objects
- Avoid requiring the ORB to maintain persistent state describing object properties
- Provide an extensible mechanism for associating policy information with objects implemented in the POA
- Allow a single servant to support multiple object identities simultaneously
- Allow multiple POAs to exist in a server
- Provide support for transient objects

Abstract POA Model:



32. Welche Aufgabe hat das CORBA Naming Service? S84

- Der wichtigste CORBA-Dienst ist der Naming Service, der Serverobjekten ermöglicht, mittels eines festgelegten Namens angesprochen zu werden. Der Namensdienst liefert dann die IOR zu einem registrierten Objektnamen. Der Naming Service ist eine Art "Telefonbuch" für Corba Objekte.
- Simple example of an object directory
- Stores a name with each object reference
- Name consists of two strings: id and kind
- Provides hierarchical naming space structure
- Hierarchy is made up of naming contexts
- Naming context can store multiple object references or other naming contexts

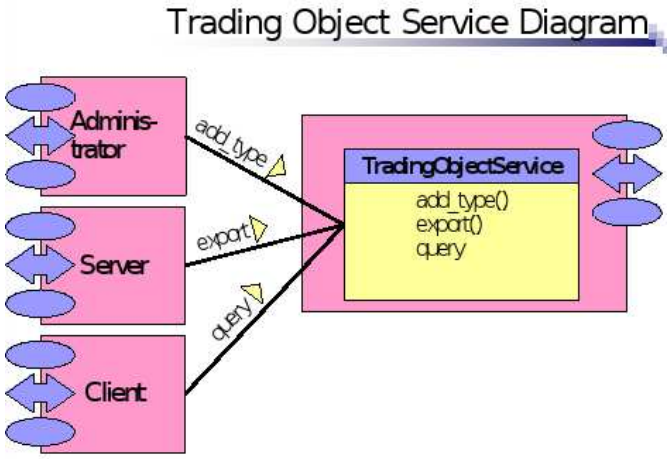


33. Welche Aufgabe hat das CORBA Trading Service? S85

- Der Trading Service ermöglicht es ebenfalls, Objekte zur Laufzeit zu finden. Allerdings werden Objekte hier über ihre Eigenschaften identifiziert und nicht durch einen Namen. Das Ergebnis einer solchen Suche können auch mehrere Objekte sein.
- Each object (offer) can have multiple properties of any type
- Provides flexible mechanism for clients to look up objects based on any subset of these properties
- Is not structured in any formal way
- Based on the concept of service type
    - contains IDL interface identifier
    - data defining the attributes that can be associated

Aufgabe:
Allows us to locate objects based on information that is important to us, rather than information that is important to the ORB

- Using Object Reference Strings
- Using Factory Objects
- Bootstrapping



34. Was ist CCM? S89

Das **CORBA Component Model** ist ein auf CORBA aufsetzendes Komponentenmodell. Die Spezifikation des CCM wurde von der Object Management Group (OMG) veröffentlicht.
Kernpunkt des CCM ist eine Komponente. Dafür führt das CORBA-Komponentenmodell den neuen Metatyp *CORBAComponent* in CORBA ein. Eine CORBA-Komponente kapselt ihren inneren Aufbau durch *Interfaces*. Diese Interfaces werden über *Ports* angeboten. Momentan wird in folgende Portarten unterschieden.

- A CORBA Component Model (CCM) application is “really” distributed
    - Could be deployed and run on several distributed nodes simultaneously
- A CORBA component could be segmented into several classes

Spezifikation:

- Abstract Component Model

    - Extensions to IDL and the object model
- Component Implementation Framework
    - Component Implementation Definition Language (CIDL)
- Component Container Programming Model
    - Component implementer and client view
    - Integration with Security, Persistence, Transactions, and Events
- Packaging and deployment facilities
- Interoperability with EJB 1.1
- Component Metadata & Metamodel

35. Was ist COM+? S94

- COM+ = Component Object Model +
- COM+ is a distributed object computing architecture for Windows.
- COM+ allows applications to communicate with one another no matter where they are located or who has designed them.
- COM+ evolved from DCOM and MTS

36. COM Anwendung S96

37. Welche Rolle gibt es beim COM Anwendungsentwicklung S98

- Design components as a client application
    - Single user, single thread, no process concern
- Write components in ANY language
    - VB, VJ, VC, Cobol, whatever
- Deploy as 1, 2, 3 or N-tier Application
    - Drag and drop simplicity

38. Welche zwei Formen von Remoting unterstützt .NET? S101

- Web Services
    - An entry point into application specified by an URL
    - Expose WebService endpoints from any process over any transport using any payload encoding
    - Process types include console apps, graphical applications, NT Services, IIS
- CLR Object Remoting
    - Builds on Web Services
    - Uses native CLR data types
    - Full CLR type system fidelity
    - Maintains distributed object identity
    - Provides object activation semantics
    - Allows control over object lifetime using leases
    - Permits out of band info using CallContext

39. Welche Vorteile hat .NET Remoting? S88? 101ff

- Could be written in different programming languages
- Could be packaged in order to be distributed
- .NET Remoting is more flexible
- .NET Remoting is customizable

40. Vergleiche die 3 Standards von CEE`S. S105 bzw Unterschied zw. RMI, CORBA und COM+? (14/19) S107,108

- Java technologies (RMI)
    - ONE LANGUAGE, MANY PLATFORMS
        - Utilise “write once, read everywhere” philosophy of Java
        - Must be implemented in Java
        - EJB standard a subset of CCM since April, 1999
- CORBA
    - MANY LANGUAGES, MANY PLATFORMS

        - Vendor-neutral specification
        - Dominates infrastructural “backbone” of distributed services
        - durch die Object Managment Group (OMG) definiert und gemanagt
- COM+
    - MANY LANGUAGES, ONE PLATFORM
        - Binary specification, language-neutral
        - Dominates desktop
        - Centre of gravity is Windows environment

	JAVA	**CORBA**	**COM+**
Protocol	JRMI utilizing Java Remote Method Protocol	IIOP	Object Remote Procedure Call
Interfaces	Home, Remote	Home, Remote, Service, Metadata	Multiple
Middleware Component Model	EJB	CCM (in CORBA 3.0)	MTS
Implementation	Java only	Binds to any conforming implementation	Language neutral

41. Was ist Flooding? S118-S120

Eine sehr einfache Möglichkeit, damit ein Router weiß, wohin er ein Datenpaket senden soll bzw wie eine Nachricht von Knoten zu Knoten gelangt ist Flooding.

- Broadcast der Nachrichten an alle Nachbarn
- einmaliges Weiterleiten aller empfangenen Datenpakete
    - bis zum Empfänger
- empfangenes Paket wird weitergesendet
    - Nachricht wird auch zurückgesendet
    - immer an alle Nachbarn
- Empfänger leitet Paket nicht mehr weiter
- andere Knoten schon
    - keine Information ob das Paket schon angekommen ist
- sehr viele Knoten empfangen Nachricht
- selbst nachdem der Empfänger die Nachricht erhalten hat, können noch Nachrichten im Netzwerk gesendet werden
    - im Beispiel nur für eine Nachricht der Fall
- jeder Knoten/Router besitzt nur sehr beschränktes Wissen

Flooding von Nachrichten:

- Sehr hoher Overhead
- zuverlässig
    - viele Pfade
    - Reliable Broadcast
- scheint wenig geeignet für Datenpakete
    - i.A. werden sehr viele Datenpakete ausgetauscht
    - Netzwerk nicht mit Datenpaketen überfluten
- Ist Flooding dennoch sinnvoll?

42. Auf welche 4 Arten kann sich die Topologie eines Netzwerks verändern? Bzw. Welche 4 Dinge können bei dynamischen Routingalgorithmen zu Änderungen in den Routingtabellen führen? S121

- Überlastung
- Mobile Knoten
- Knoten kommen hinzu, weg, fallen aus,
- neue Verbindungen, Segmentierung in Teilnetze

43. Welche Routingtabellen werden erstellt? S122

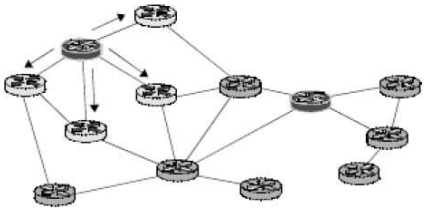
- statische (Pfade fix vorgegeben/Veränderung nicht automatisch berücksichtigt)

- halbdynamische (Tabs werden vorausberechnet/zusätzl.Änderungen werden erkannt)
- dynamische (Nachbarknoten bekannt/Wissen über Topologie von Grund auf / Änderungen werden erkannt)

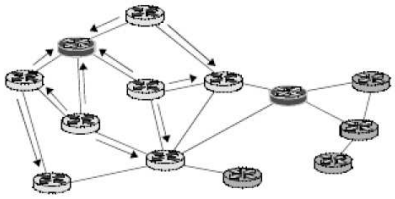
Routingtabellen spiegeln den Zustand des Netzes wider. Können fortlaufend aktualisiert werden (Proaktiv) oder bei Bedarf erstellt werden (Reaktiv)

44. Beschreiben Sie den Flooding Algorithmus zum Austausch von Informationen. S124

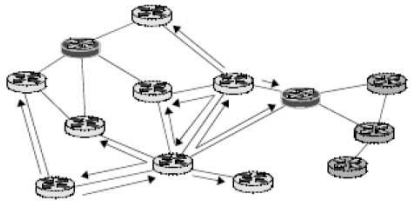
- Broadcast der Nachrichten an alle Nachbarn
- einmaliges Weiterleiten aller empfangenen Datenpakete
    - bis zum Empfänger



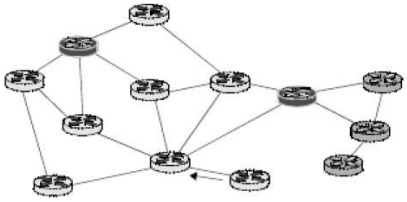
- empfangens Paket wird weitergesendet
    - Nachricht wird auch zurückgesendet
    - immer an alle Nachbarn



- Empfänger leitet Paket nicht mehr weiter
- andere Knoten schon
    - keine Information ob das Paket schon angekommen ist
    - sehr viele Knoten empfangen Nachricht



- selbst nachdem der Empfänger die Nachricht erhalten hat, können noch Nachrichten im Netzwerk gesendet werden
    - im Beispiel nur für eine Nachricht der Fall
- jeder Knoten/Router besitzt nur ein beschränktes Wissen



Eigenschaften:

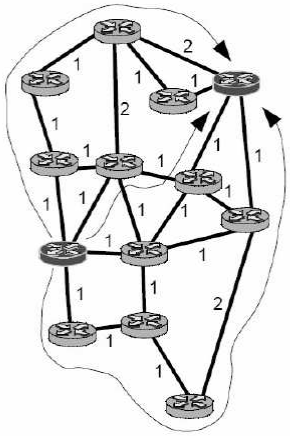
- Sehr hoher Overhead
- Zuverlässig
- Scheint wenig geeignet für Datenpakete

45. 4 Anforderungen an den Routing-Algorithmus S126

- Packet Delay (eines Packets)
- Energieverbrauch/ Verzögerung (eines Knotens)
- Durchsatz / Packet Loss (aller Pakete)
- Konvergenz (Fähigkeit eines Routing Algorithmus Änderungen in der Netzwerkstruktur stabil bei die Entscheidungsfindung zu berücksichtigen)
  (alle Knoten)

46. Was macht ein Metrik S127

Eine **Routing-Metrik** ist ein Wert, mit dessen Hilfe ein Routing-Algorithmus feststellen kann, ob eine Route im Vergleich zu einer anderen besser ist (Bei mehreren möglichen Routen wird eine Route mit kleiner Distanz im Sinne der Metrik bevorzugt.).
Metriken können Informationen wie z. B. Bandbreite, Verzögerung, Hop Count, Pfadkosten, Last, MTU, Verlässlichkeit und Kommunikationskosten berücksichtigen.



- jede Verbindung wird bemessen
- Routingalgorithmus wertet diese Maße aus (Metrik)
- unterschiedliche Routen werden vergleichbar
- Optimalität
- teilweise abstrakte Kosten
    - Entfernung (Hop Count)
    - Reale finanzielle Kosten
    - Netzwerklast
    - Verzögerung
    - Bandbreite
    - ...

47. Nennen Sie 4 verschiedene Arten von Metriken. S127

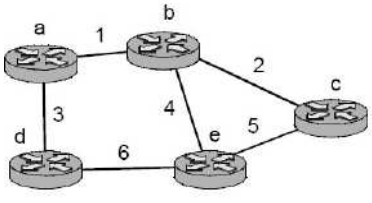
- Entfernung (Hop Count)
- Reale finanzielle Kosten
- Netzwerklast
- Verzögerung
- Bandbreite

48. Warum verwendet Link State/E.W.Dejkstra SPF S135,136

Um aus den Daten der Datenbank den kürzesten Weg zu erhalten (S.136 unten)

OSPF ist ein dynamisches Routing-Protokoll innerhalb eines autonomen Systems. Es hat das Routing Information Protocol (RIP) als das Standard Interior Gateway Protocol (IGP) abgelöst, insbesondere bei großen Netzen. OSPF verwendet die Kosten eines Pfades als Metrik und kann bei gleichen Kosten lastverteilt arbeiten. Kosten werden bei OSPF standardmäßig aus der verfügbaren Bandbreite berechnet. Jeder Knoten fügt die empfangenen Informationen der Link State Pakete zu einer Tabelle zusammen dieselbe Tabelle in allen Knoten.
Mit Hilfe dieser Daten wird der kürzeste Pfad berechnet

- Shortest Path First
- E.W.Deijkstra



Von	Nach	Verbindung	Entfernung
a	b	1	1
a	d	3	1
b	a	1	1
b	c	2	1
b	e	4	1
c	b	2	1
c	e	5	1
d	a	3	1
d	e	6	1
e	b	4	1
e	c	5	1
e	d	6	1

E.W.Dijkstra:
Der Algorithmus von Dijkstra dient der Berechnung eines kürzesten Pfades zwischen einem Startknoten und einem beliebigen Knoten in einem kantengewichteten Graphen.

Merkmale:

- OSPF garantiert ein schleifenfreies Routing im Gegensatz zu RIP (= verhindert Kreisrouting)
- Hello-Protokoll für die Überwachung der Nachbarn
- OSPF ist für große skalierbare Netze gut geeignet

- Das Area-Konzept vereinfacht die Kommunikation und Wartung
- OSPF ist ein offener Standard

49. Nennen Sie 4 Anforderungen, die Routing in (mobilen) Adhoc-Netzwerken von kabelgebundenen Netzwerken unterscheidet. S148, 149

- lokaler Broadcast
- hoher Packet Loss
- geringe Bandbreite
- hohe Mobilität
  - Routenänderung
  - Verbindungsverlust
- Energieversorgung
  - Batterie
- Sicherheit (Security)
- asymetrische Verbindungen
- nicht alle Eigenschaften treffen auf alle MANETs zu

50. Welche Eigenschaften gibt es beim Distanzvektorprotokoll? + Beispiel S150, 151

- Schleifenfreiheit
- Routingtab. mit Anzahl Hops zum Ziel/Sequenznummer /Route des kürzesten Wegs
- periodisches Tabellenupdate

- einfache Berechnungen+ Implementierung
- skaliert schlecht
- viele Nachrichten (Periodisch/bei Änderungen)
- verzögertes Routing Update (Zeitmessung bis Route persistent ist/ Settling Time/ dämpfende Eigenschaften)

51. Welches Betriebssystem befindet sich am LonWorks NEURON Chip? S168,169

„Scheduler“ oder „NEURON Chip Firmware“

- Neuron Scheduler → „Round Robin“

52. Welche Aufgabe hat das Node-Objekt in einem LonMark-konformen LonWorks-Knoten? S173, 174

Definieren von NV’s und Configuration Properties, Node Selfdocumentation
e.g. typical Node-Object = “Neuron Chip”

53. Neuron chip, welche OS S174

siehe Frage oben.

54. Welche Porte verwendet der NEURON Chip? Wozu?

input output port und einen netzwerkport ...
ich schätz vom input output krieg bzw. schickt er infos an die schnittstelle und der netzwerkport verbindet ihn mit der außenwelt ...

55. Wie heisse die 3 Teile einer LonWorks NEURON CPU? Seitenangabe

- 3 Prozessoren (MAC, Network, Application)
- ALU
- Network und Application Buffers

56. Was ist ein LonMark-Objekt?(2/19) S173,174

An object defines

- what NVs are where (NV#)
- how the NVs are called
- what NVs do what (semantics)
- behavior of the program that implements this object (but not the source code!)

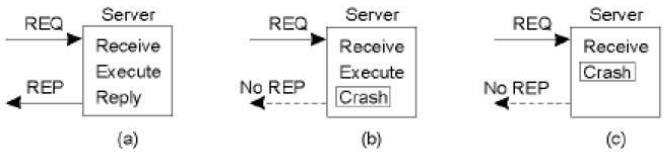
57. Zählen sie 3 Arten von Fehler bezüglich deren Auftretens auf! S202ff

- Failure (Fehlfunktion) A system is said to fail when it cannot meet its promises. (E.g. a distributed system cannot provide one or more services completely.)
- Error (Fehlerzustand) An error is part of a system.s state that may lead to an failure
- Fault (Fehlerursache) The cause of an error is called fault..

Oder
Software (Spec., Impl., fault) und Hardware-Fehler (Spec., Component malfunction fault)

58. Wie kann man den Fehler "The request message from the client to the server is lost" am besten erkennen und beheben? S206

Fehler erkennbar aufgrund des Failure Models. → in diesem Fall: Omission failure
Lost request messages Server Crashes



A server in client-server communication
b) Normal case
c) Crash after execution
d) Crash before execution

59. Was ist die Besonderheit am menschlichen Körper gegenüber einem technischen System im Bezug auf die Verfügbarkeit und Zuverlässigkeit seiner Komponenten? S217

- Availability (Verfügbarkeit): The organic system have ANY TIME in state that is capable to guarantee life functions.
- Reliability (Zuverlässigkeit): Essential functions have ALLWAYS be guaranteed.

The human body has to be more reliable and available than its components.

60. Faulttolerance auf Zellebene? S219

- Massive redundancy
  - Of cells and cell groups
- Preemptive replacement (präventive Erneuerung)

61. Was sind 4 grundlegende Funktionen eines System-Agents? S223

Ein Softwareprogramm das:

- Ziele hat
- unabhängig reagiert um es zu erreichen
- Strategien und Pläne entwickelt, um
- mit anderen Agenten zu kooperieren

input → intelligent behaviour → output

62. Welche Eigenschaften ein Programm besitzen, um eine intelligenter Agent zu sein S223, 224

siehe Frage oben.

63. Erklären Sie den prinzipiellen Timeraufbau (5/19) S230

- Node(s) with accurate clocks have to be designated
- Such a node distributes its current time regularly (Time Sync Packet)
- Every node samples its local time at which such a packet is received (Time stamps it!)
- The absolute transmission delay has to be known (measured periodically)
- Deviation can be calculated and corrected

64. Welche 3 Parameter beeinflussen die netzwerkbasierende Zeitsynchronisation und wodurch werden sie verursacht? S232 bzw 235

Ähm das packerl wird ja ins netzwerk geschickt, also ist auf jeden fall einmal der netzwerk parameter da. nämlich:

das schicken des pakets generell, also die cpu ruft die netzwerkkarte (vereinfacht gesagt) und übermittelt die daten dorthin (also verzögerung)

die netzwerkkarte kriegt das zeitpaket, und muss jetz mal abchecken an wen das gehen soll und ob überhaupt a platz in der leitung dafür ist, also wieder verzögerung, und weiters dann der datenstau in der leitung selbst.

natürlich ist das ganze beim client der das paket bekommt nicht anders, der muss es aufnehmen und - sofern die cpu nicht zu beschäftigt ist der cpu übergeben damit die die zeit dazu veranlassen kann sync zu werden.

ich hoff das is nicht vollkommener schwachsinn aber so ca stehts im skriptum...

edit: ich glaub abhilfe der verzögerungen schafft man durch ein CSP (siehe hardware Timestamp) - da bin ich mir aber net ganz sicher - wird der zeitstempel in pakete beim versenden reinkopiert oder in pakete die versandt werden?

genauigkeit / intervall für ungenauigkeit / networkload overhead
(siehe seite 235)
wodurch:
- delay for completing packet by cpu
- delay for network controller being available
- delay how long channel is busy
- delay of transmitting

und beim receiver natürlich in umgekehrter richtung
(also eh im grund wie du es geschrieben hast)

65. Was versteht man unter Hardware Time-Stamping? (6/19) S233

- Time stamp - Every node samples its local time at which such a packet is received (Time stamps it!)
- Hardware Time – Stamp → Hardware generated time stamps are copied into the packet while it is actually transmitted (received)

66. Welches sind die 6 wichtigsten Sicherheitsaspekte (6/19) S251

1. Confidentiality
2. Integrity
3. Availability
4. Authentification
5. Access Control
6. Non-Repudiation

Bei 1. bis 3. spricht man auch von CIA.

67. Wodurch zeichnen sich symmetrische kryptographische Algorithmen aus? Geben Sie 2 Beispiele für solche Algorithmen. S255

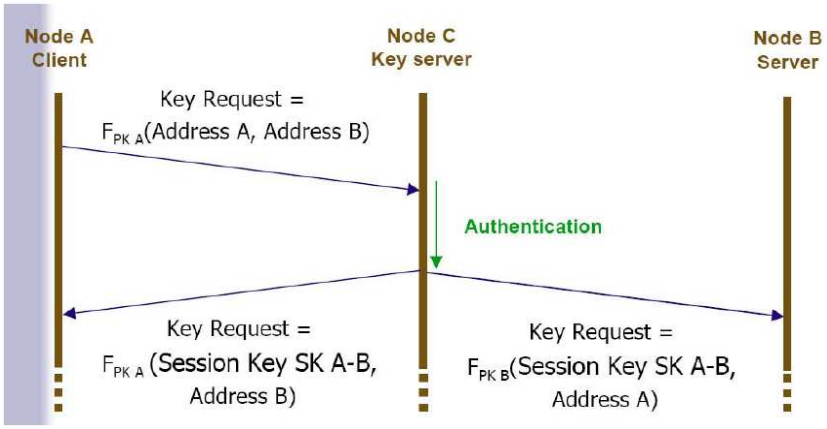
- Blockchiffren sind symmetrische Verfahren => Sender und Empfänger nutzen gemeinsamen Schlüssel
- Blockchiffren beschreiben Verschlüsselung eines Blocks
- Zusammenspiel bei mehreren Blöcken: Verarbeitungsmodi (= Grundchiffre + Rückkoppelung + einfache Operationen)
- Verschlüsseln in 8 Byte Blocklänge
- Vier Modi für Blockchiffren: Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB) und Output Feedback (OFB)

Bsp1: Data Encryption Standard (DES) mit 64Bit Blocklänge

Bsp2: Nachfolger von DES –> Advanced Encription Standard (AES), mind 128bit Blocklänge

68. Wie wird in BACnet ein Sessionkey zwischen zwei Knoten ausgetauscht? (3/19) S261

BACnet = Building Automation and Control Networks ist ein Netzwerkprotokoll für die Gebäudeautomation.



69. Schlüsselverteilung im BACnet? S263

- Initial key distribution – plain text
- Consecutive key distribution
  - Increment to the key (update key)
- Nur ein Schlüssel pro Knoten
- Symmetrischer 48 bit Schlüssel
- Vital areas of the Node are not protected
  - SNVT2 structures and the application data area can be read/written over the network

71. Geben Sie ein Beispiel eines permanenten Fehlers

Software Bug, burnout Chips, disc head crashers,
Ein permanenter Fehler existiert solange bis die fehlerhafte Komponente repariert wurde.

72. Eine Link State Routing Tabelle für einen Knoten angeben (wie im Skriptum)

73. CORBA Struktur beschreiben anhand von den Komponenten

- A distributed component-oriented model
  An architecture for defining components and their interactions
  From client-side (GUI) to server-side (business) components
- A packaging technology for deploying binary multi-lingual exexutables
- A container framework for injecting lifecycle, (de)activation, security, transactions, persistence, and events
- Interoperability with Enterprise Java Beans (EJB)

74. EJB beschreiben anhand der Struktur/Komponenten

SCHON BEANTWORTET

75. Nennen Sie die vier prinzipiellen Strukturen von DCS (decentralized computing systems)

77. Welche Arten von Fehlern gibt es (3)?

1. Fail
2. Error
3. fault

78. Wie kann man Chipkarten einteilen?

- Chip-Technologie
  1. Speicherkarten
  2. Prozessorkarten
- Kommunikations-Technologie
  1. Kontaktkarten
  2. Kontaktlosekarten
  3. USB-Karten

79. Welche Arten von DCS (Distributed Computing Systems) gibt es ?

1. Minicomputer
2. Workstation
3. Workstation-Server
4. Hybrid
5. Pool of Processors

80. Welche 2 prinzipiellen Arten von Uhren (Timern?) gibt es? (Seite 79)

Counter based Clock
Adder based Clock

81.Was beschreibt das Atomic Multicast Problem?

Es wird gefordert, dass alle Nachrichten in derselben Reihenfolge wie alle Prozesse geliefert werden. An den Server werden entweder alle Nachrichten geliefert oder gar keines.

82. Netz mit 5 Knoten gegeben, man soll für einen Knoten eine LinkState Routing Tabelle erstellen.

83. Welche CPUs hat der Neuron Chip und was machen sie? (ab S.141)

- MAC Processor
  Media Access Control
  Layer 1 and 2
- Network Processor
  Layers 3 - 6
- Application Processor
  Layer 7 and Application

84. Was bedeutet "Marshaling" und was sind Proxys? S.177

Marshalling- wandelt Datenelemente in ein Format um, um sie in einer Nachricht an den Empfänger zu schicken (Bsp.: RMI, Stub/Skeleton)
Transmitting req. And resp.
Proxy – Füllt die Lücke zwischen remote und local object references
Client/Server – side proxies transmit requests and responses