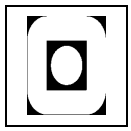
NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for informasjonsteknologi, matematikk og elektroteknikk

Institutt for datateknikk og informasjonsvitenskap



Løsning på eksamen i TDT4190 Distribuerte systemer
Torsdag 9. juni 2005, 0900-1300

Det ønskes korte og konsise svar på hver av oppgavene. Det vesentlige er å kunne dokumentere forståelse, beherske prinsipper og se sammenhenger - ikke å kunne gjengi en mengde detaljer.

Der det synes å mangle noen opplysninger, må det angis hvilke antagelser som synes å være naturlige. Merk at viktige begreper er angitt på både norsk og engelsk.

Oppgave 1 – Modeller og standarder (Models and standards) – 12.5 %

a) Angi kort hva vi bruker mellomvare (middleware) til

SVAR:

En konstruerer et programvare lag (med et generelt API) mellom globale applikasjoner og lokale ressurser slik at en gitt applikasjon kan nå en vilkårlig lokal ressurs, og en gitt ressurs kan nås av en vilkårlig applikasjon – hvor mellomvare laget står for ruting av forespørslene til riktig node og oversetting av forespørslene til riktig språk

b) Beskriv hovedprinsippene i OMGs CORBA (Common Object Request Broker Architecture)

SVAR:

Mellomvare-arkitektur – ORB-type

CORBA-objekter
- Applikasjonsobjekt – Felles fasiliteter – Objekttjenester

ORB-komponenter
- Klientobjekter – Objektimplementeringer
- IDL stubb – IDL skjelett
- ORB-kjerne – ORB grensesnitt – Objekttilpasning
- Dynamisk innvokerings-grensesnitt – Dynamisk skjelett-grensesnitt
- Grensesnittlager – Implementasjonslager

Interoperabilitets-protokoller/-broer
- GIOP/IIOP, GIOP/OSI-GIOP, ESIOP/DCE-CIOP
- OSI-GIOP – IIOP, DCE-CIOP – IIOP, DCE-CIOP – OSI-GIOP

Oppgave 2 – Kommunikasjon og synkronisering (Communication and synchronization) – 12.5 %

a) Angi kort hva vi trenger henholdsvis logiske klokker (logical clocks) og vektorklokker (vector clocks) til

SVAR:

Logiske klokker og vektorklokker brukes til å implementere et felles tidsbegrep

Logisk klokker holder oversikt over hvor mange hendelser på alle nodene samlet som er kjent på en gitt node

Vektorklokker holder oversikt over hvor mange hendelser i hver enkelt node som er kjent på en gitt node

Vektorklokker – ut over logiske klokker – skiller mellom hendelser på ulike noder

b) Beskriv en algoritme for distribuert primas utvelgelse (distributed process election)

SVAR:

Av de tre nedenfor viste varianter (enkel / avansert / typisk) holder det med en - helst den typiske eller den avanserte, helst ikke den enkle

- *Sentralisert algoritme (enkel)*

Start Valg-rutine: 1a) Send Valg-melding til global koordinator

Får Valg-melding: 2a) Send N-1 * OK?-melding 2b) Motta ≤ N-1 * OK!-melding 2c) Send Sjef-melding til alle m/ Størst identifikator

Får OK?-melding: 3a) Send OK!-melding til global koordinator

- *Distribuert algoritme (avansert)*

Start Valg-rutine: 1a) Send Valg-melding til alle med høyere identifikator

Får Valg-melding (fra noen lavere): 2a) Send OK-melding til avsender 2b) Start Valg-prosess som over

Får OK-melding (fra noen høyere): 3a) Gi opp for andre

Får ingen melding (fra noen høyere): 4a) Send Sjef-melding til alle m/Egen identifikator

- *Ringalgoritme (typisk)*

Start Valg-rutine: 1a) Send Valg-melding m/ Egen identifikator

Får Valg-melding: 2a) Med høyere identifikator og Egen ikke inkludert / Egen inkludert: Send videre m/ Gitt identifikator 2b) Med lavere identifikator og Egen ikke inkludert: Send videre m/ Egen identifikator 2c) Med lavere identifikator og Egen inkludert: Stopp Valg-melding 2d) Med samme identifikator og Egen inkludert: Send Sjef-melding m/ Egen identifikator

Får Sjef-melding: 3a) Med høyere identifikator: Send videre m/ Gitt identifikator 3b) Med samme identifikator: Avslutt rutine

Oppgave 3 – Distribuerte filsystemer (Distributed file systems) – 12.5 %

a) Angi kort hva vi bruker henholdsvis caching (caching) og replisering (replication) til

SVAR:

Caching og replisering innfører kopier av et dataelement nærmere bruksstedene enn det dataelementet i seg selv tilsier

Caching tilbyr en forholdsvis temporær kopi
Replisering tilbyr en mer permanent kopi

Disse kopiene – permanent i større grad enn temporær – tilsier også økt sikkerhet i tillegg til økt nærhet

b) Beskriv hovedprinsippene i SUNs NFS (Network File System)

SUN NFS	
Globalt filsystem:	Ikke felles
Cache type:	Enkelte blokker
Konsistens:	Sjekking
Oppdat. Semantikk:	Aksjon
Replikat type:	Begrenset
Orientering:	Feiltoleranse
Implementasjon:	Kjernenivå

Oppgave 4 – Distribuerte databasesystemer (Distributed database systems) – 12.5 %

a) Angi kort hva vi trenger 2-fase-låsing (2 phase lock) til

SVAR:

2PL brukes til å forhindre gale resultater i.f.m. at flere applikasjoner aksesserer og/eller endrer data samtidig

Inkonsistente uthentinger og tapte oppdateringer er typiske eksempler på dette

b) Beskriv hvordan 3-fase-bekrefting (3 phase commit) virker

SVAR:

3PC innfører en tredelt avslutning: 1.fase m/stemming & 3. fase m/utføring –
etter alle-har-veto prinsippet koplet til enten feil eller autonomi, med en ekstra mellomfase (i forhold til 2PC) hvor nodene blir forberedt på det endelige resultatet gitt at ingen feil oppstår

3PC sikrer autonomi og effektivitet med mye overhead for unntakstilfeller med mye feil

Oppgave 5 – Distribuert pålitelighet (Distributed reliability) – 12.5 %

a) Angi kort hvilke utfordringer replisering (replication) og / eller kommunikasjonsfeil (communication errors) gir i distribuerte databasesystemer (distributed database systems)

SVAR:

Uten replisering – Uten kommunikasjonsfeil:
- Ingen store utfordringer

Uten replisering – Med kommunikasjonsfeil:
- 3PC-terminering av transaksjoner kan gi blokkering i forsøk på å oppnå atomiskhet

Med replisering – Uten kommunikasjonsfeil:

- 2PL-låsing for transaksjoner må få utvidelser for å sikre kollisjon av tilhørende låser

Med replisering – Med kommunikasjonsfeil:

- MajKonsensus- / VirtPartisjon-algoritmer må til for å utpeke partisjon for oppdatering
- 2PL-låsing for transaksjoner må få utvidelser for å sikre kollisjon av tilhørende låser
- 3PC-terminering av transaksjoner kan gi blokkering i forsøk på å oppnå atomiskhet

b) Beskriv en algoritme for distribuert oppdatering (distributed updating) hvor både replisering (replication) benyttes og kommunikasjonsfeil (communication errors) forekommer

SVAR:

Av de tre nedenfor viste varianter (1 enkel / 2 avanserte) holder det med en, og til og med den enkle – selv om den ikke er en komplett løsning

- *Tilgjengelige kopier (enkel – kun serialiserbarhet per node)*

Aksessering underveis: Les og R-lås en tilgjengelig kopi ! Skriv og W-lås alle tilgjengelige kopier !

Validering til slutt: Er alle tidligere tilgjengelige kopier fortsatt tilgjengelige ? Er alle tidligere utilgjengelig kopier fortsatt utilgjengelige ?

- *Majoritets konsensus (avansert – også serialiserbarhet mellom noder)*

Uthenting: Les og R-lås en majoritet av tilgjengelig kopier Returner den kopi med høyeste versjonsnummer

Oppdatering: Skriv og W-lås en majoritet av tilgjengelig kopier Inkluder i kopi et høyere versjonsnummer

- *Virtuelle partisjoner (avansert – også serialiserbarhet mellom noder)*

Kontroller etter hver tilstandsendring: Sikre tilgang til en majoritet av lesbare kopier i løpende tilstand Sikre tilgang til en majoritet av skrivbare kopier i løpende tilstand

Aksesser mellom tilstandsendringer: Les og R-lås en kopi i løpende tilstand Skriv og W-lås alle kopier i løpende tilstand

Oppgave 6 – Distribuerte navnetjenester (Distributed name services) – 12.5 %

a) Angi kort hvordan vi kan utnytte caching (caching) og replisering (replication) i slike tjenester i forhold til hvordan de må benyttes i distribuerte filsystemer (distributed file systems) generelt

SVAR:

I navnetjenester tilbyr caching og replisering kopier av pekerinformasjon
I filsystemer generelt tilbyr caching og replisering kopier av verdiinformasjon

Om pekerinformasjon er gyldig eller ikke oppdages ved å forfølge pekerne
Om verdiinformasjon er gyldig eller ikke avdekkes kun ved en sjekk mot originalen

Navnetjenester trenger således i mye mindre grad enn filsystemer generelt kontinuerlig å kontrollere gyldigheten av sin informasjon

b) Beskriv hovedprinsippene i DNS (Domain Name System)

SVAR:

- *DNS*
  Brukes i Internett – med et sært navnerom
  Brukes spesielt for maskinnavn og postnavn

Oppgave 7 – Distribuert delt lager (Distributed shared memory) – 12.5 %

a) Angi kort forskjellene mellom gradsavballansert konsistens (degrees of consistency) og tidsavballansert konsistens (times of consistency)

SVAR:

Gradsavballansering:

- Kopling av konsistens til anvendelser ved å avgjøre i hvilken grad DSM må være konsistent

  Prosessor konsistens:
  Alle prosessorer ser lokale operasjoner i samme rekkefølge

  Kausal konsistens:
  Alle prosessorer ser lokale og koplete, globale operasjoner i samme rekkefølge

  Sekvensiell konsistens:
  Alle prosessorer ser lokale og globale operasjoner i samme rekkefølge

Altså gradvis tyngre krav: PK => KK => SK

Tidsavballansering:

- Kopling av konsistens til synkronisering ved avgjøre til hvilke tider DSM må være konsistent

  Båndleggingskonsistens:
  Fellesdata gjøres konsistente før kritiske regioner

  Frigjøringskonsistens:
  Fellesdata gjøres konsistente etter kritiske regioner

  Svak konsistens:
  Fellesdata gjøres konsistente både før og etter kritiske regioner

  Altså ikke helt gradvis tyngre krav: BK => SK & FK => SK

b) Beskriv hovedprinsippene i MUNIN og MIDWAY

SVAR:

MUNIN:

- Kjøretidssystem simulering av delt lager (dvs. postbasert)
- Variabelbasert overføring (altså ikke objektbasert)
- Frigjørings konsistens (som tidsavballansering)

MIDWAY:

- Kjøretidssystem simulering av delt lager (dvs. postbasert)
- Variabelbasert overføring (altså ikke objektbasert)
- Båndleggings konsistens (som tidsavballansering)

Oppgave 8 – Distribuerte multimediasystemer (Distributed multimedia systems) – 12.5 %

a) Angi kort hvordan distribuerte multimediasystemer skiller seg fra andre distribuerte systemer (distributed systems in general)

SVAR:

- Store mengder med kontinuerlige data omfattes
- Korte frister eksisterer, og små variasjoner tillates, ved levering
- Svært korte tur-retur tider kreves for interaktive MM-applikasjoner
- Store, skiftende ressurskrav som ikke kan monopolisere parallelt kjørende applikasjoner

b) Beskriv hvilke utfordringer det gir å designe og implementere tjenestekvalitet (quality of service) for slike distribuerte systemer

SVAR:

- Tilgangskontroll: Må kontrollere ressurskrav opp mot ressurstilbud på sanntidsvis
- QoS-forhandlinger: Vil tilby forhandlinger i.f.m. oppstart og endring av ressursbehov
- Ressursforvaltning: Vil garantere tilgjengelighet av ressurser for aksepterte applikasjoner
- Status: Må basere seg på liten QoS-funksjonalitet i dagens internett-teknologi