Afdeling Informatica	**Gespreide Systemen**
Vrije Universiteit	**20.12.2005**

Deel I

Dit deel beslaat dezelfde stof als dat van de toets.

1a Leg de principiële werking van een *remote procedure call (RPC)* uit. *5pt*

1b Geef twee overtuigende argumenten waarom een RPC nooit dezelfde semantiek kan bieden als een lokale procedure aanroep. *5pt*

1c Wat is het grootste verschil tussen een *remote method invocation (RMI)* en een RPC? *5pt*

2a Beschouw een file server die een tabel bijhoudt van (client,file) paren die elke aangeven welke client welke file gevraagd heeft. Is deze server *stateful*? Motiveer je antwoord. *5pt*

2b Leg uit wat een *message-queuing server* is en of het als een *stateless server* ontworpen kan worden. *5pt*

2c Het X Window systeem refereert aan de X kernel op de client machine als de "X server", en de applicatie die op de server machine draait, als de "X client". Waarom is dit feitelijk correct gebruik van client/server terminologie? *5pt*

3a Beschouw een client die z'n klok elke minuut tracht te synchroniseren met die van een *time server*. Daartoe zendt het een aantal verzoeken, waarvan de resultaten hieronder staan. Hoe zal de client zijn klok aanpassen bij de ontvangst van de server's antwoord? Tijd is gegeven in (uur:min:sec:msec). De verwerkingstijd bij de server is nul. *5pt*

Sent at (local time)	**Round-trip delay**	**Response**
10:54:00:00	18 ms	10:54:00:10
10:55:00:00	24 ms	10:55:00:12
10:56:00:00	22 ms	10:55:00:10

3b Leg het principe van het Berkeley algoritme uit om de klokken aan te passen in een gedistribueerd systeem. *5pt*

3c De communicatielaag in een gedistribueerd systeem kan bijhouden welke berichten causaal gerelateerd zijn. Wat zijn de twee belangrijkste bezwaren tegen deze functionaliteit? *5pt*

Deel II

4a Is de onderstaande volgorden van gebeurtenissen toegestaan bij een sequentieel-consistente opslag? En bij een causaal-consistente opslag? Motiveer je antwoord. *5pt*

P1:	W(x)a			W(x)c	
P2:		R(x)a	W(x)b		
P3:		R(x)a		R(x)c	R(x)b
P4:		R(x)a		R(x)b	R(x)c

4b Leg uit waarom *writes-follow-reads* consistentie garandeert dat causale relaties gehandhaafd blijven wanneer het toegepast wordt bij de implementatie van een gedistribueerde newssysteem. *5pt*

4c Wat zijn de voorwaarden om *read-write* en *write-write* conflicten in een quorum gebaseerd systeem te voorkomen? *5pt*

5a Laat zien dat 4 processen waarvan één zich misdraagt, dat de 3 overige processen altijd tot overeenstemming kunnen komen ongeacht wat er gecommuniceerd wordt door het foute proces. *5pt*

5b Beschouw een print server met drie mogelijke gebeurtenissen: (M) vertel de client dat printen klaar is; (P) print; (C) crash. Als een client een bericht krijgt dat de server net hersteld is van een crash, volgt een van de vier mogelijke strategiën: (1) Nooit een printverzoek opnieuw indienen; (2) Altijd het verzoek opnieuw indienen; (3) Alleen opnieuw indienen als de ontvangst van het vorige verzoek bevestigd is; (4) Alleen opnieuw indienen als er geen bevestiging binnengekomen is dat het vorige verzoek afgeleverd is. Laat zien dat als de server altijd het printen bevestigt na het printen klaar is, het onmogelijk is om een schema te bedenken waarbij het printen niet verloren raakt, of dubbel uitgevoerd wordt. *10pt*

6a NFS is beslist niet een filesysteem. Leg uit waarom niet. *5pt*

6b Coda staat toe dat clients files in een lokale cache opslaan, maar zal een notificatie sturen als een file gewijzigd wordt. Wat is de belangrijkste reden om deze *callbacks* parallel uit te voeren? *5pt*

6c Coda staat toe dat een (lezende) client gebruik kan blijven maken van zijn lokaal gecached file, zelfs als er gelijktijdig wijzigingen plaatsvinden bij de server. Waarom wordt dit gedrag als correct beschouwd? *5pt*

Cijferbepaling: *(1) Tel per deel de behaalde punten bij elkaar op. (2) Laat T het totaal aantal behaalde punten voor de toets zijn ($0 \leq T \leq 45$); $D1$ het aantal behaalde punten voor deel 1; $D2$ het aantal behaalde punten voor deel 2. Het eindtotaal E is dan* $\max\{T, D1\} + D2 + 10$.