Prüfungsfragen Betriebssyteme 1999

BP I, AKBP
Hofmann
April 1999

Bemerkungen zu Prüfung und Prüfer

- Schwerpunktfach
- Ergebnis: 2.3

Fragen

- Verteilte Systeme, wie funktioniert die Kommunikation zwischen Rechnern?
RPC, sockets, namedpipes, messages, virtual shared memory...
Irgendwie wollte er wohl auch was von paketorientierten und stromorientierten Verbindungen hören, hat er mir dann aber auch gesagt.
- Und sonstige (alle hinschreiben)
exitstatus, Dateien, shared memory, signals,...
- Zu all diesen Mechanismen,
1) Adressierung,
2) Einheit, die verschickt wird,
3) Flusskontrolle ?
z.B. pipes: 1)Dateiname, 2)Byteweises lesen, 3)??,
sockets: 1)Dateiname oder IP-Adresse und Port, 2)Pakete begrenzt durch HW??, 3) bei TCP/IP
- Dazu genauer: Warum ist IP unzuverlässig, auch wenn das Kommunikationsmedium zuverlässig ist? (ich kam nicht drauf, Hinweise: Ethernet, wie wird da etwas entgegegenommen: Ethernetcontoller, Etherkontrollblock)
HW kann evtl. nicht so schnell annehmen, wie gesendet wird
- Wie groß darf ein IP-Paket sein?
Begrenzt durch HW???
- Was steht im header des IP-Pakets?
- Wofür braucht man die max. Anzahl an hops?
Paket geht verloren, irrt umher, ICMP-Nachricht, dass es nicht angekommen ist
- Für was taugt UDP/IP überhaupt, wenn es doch unsicher ist
Verwendung in RPC, etc. D.h höhere Mechanismen, die sich selbst drum kümmern
- Was macht TCP/IP sicher
Flusskontrolle: sliding window - alternating bit protocol
- Sockets genauer
Unix domain, Internet domain, stream sockets, datagramm socket,...
- Unterschiede von RPCs mit TCP/IP, UDP/IP

BP I, AKBP
Hofmann
Juli 1999

Bemerkungen zu Prüfung und Prüfer

- Meine Prüfung ging über IPC auf einem Rechner bis über IPC über ein Kommunikationssystem und dem höheren Konzept des Fernaufrufs
- Sowohl auswendiggelerntes Wissen als auch Denken war nötig, alles wurde in einen Zusammenhang gesetzt und deswegen war es relativ einfach, die Antworten zu finden.
- Verständnis reichte vollkommen und Note war fair.

Fragen

- IPC Mechanismen + kurz erklären
(Sockets, Messages, Pipes,... - jeweils in einem Satz erläutert + Sockets genauer (Datagramm, Stream, Raw, Kern, Seqpacket (jeweils ein Satz dazu)
- Kann auch auf einem Rechner Datenverlust bei Sockets auftreten?
(ja, begrenzter Puffer)
- Welche IPC-Mechanismen sinnvoll bei verteiltem System?
(Sockets, named Pipes, Messages, Dateien, shared Memory (virtuelles), bei verteilten Prozessen auch exit-Status und Signale sinnvoll)

- Neuere Unix-Systeme, wie werden Pipes implementiert?
(Hofmann: mit Sockets)

- Wie würden Sie Sockets implementieren?
(Protokolle - entweder TCP/IP oder UDP/IP)

- Unterschied TCP/UDP, wofür braucht man IP?
(zuverlässig- unzuverlässig, IP: Portadressierung)

- Zuverlässiges Kommunikationssystem: Wo gehen bei IP Pakete verloren?
(Ein Rechner kann nicht schnell genug annehmen)

- Wie TCP zuverlässig?
(Fenstergröße, kein Alternating-Bit-Protokoll)

- Fernaufruf?
(At-least-one-Semantik, wann ausreichend - Abfragen und Transaktionen, wie last-of-many? Client muss veraltetes Ergebnis erkennen → Sequenznummern)

- Verwaiste Aufrufe
(Expiration, Extermination..., warum nicht Extermination? Mehrere Rechnerausfälle, also: Mischform)

OODS
Kleinöder, Beisitzer: Golm
Juli 1999

Bemerkungen zu Prüfung und Prüfer

- Scheinkolloquium

- Ergebnis: 1,3

- Faire Bewertung (allerdings gehen auch die sehr aufwendigen Programmieraufgaben in die Note mit ein)

- gibt Hilfestellungen

- Man solltea auch über die Zusammenhänge der Teilbereiche Bescheid wissen. (z.B. Welche Vorteile bringt Objektorientierte Programmierung bei Verteilten Systemen)

Fragen

- Was haben wir in der Vorlesung eigentlich gemacht?

- Was sind die Grundkonzepte der objektorientierten Programmierung? (ich habe genannt: Objekte, Klassen, Hierarchiebildung, Information Hiding)

- Welche Vorteile bietet objektorientierte Programmierung bei Verteilten Systemen? z.B. Verlagerung von Objekten.

- Warum ist es einfacher ein Objekt zu verlagern als einen Prozess?

- Wie funktioniert RMI in CORBA bzw. in Java?

- Welche Fehler können beim RPC auftreten?

- Können diese Fehler auch auftreten, wenn man TPC benutzt?

- Was benutzt man bei RPC anstelle exactly-once-Semantik? at-most-once-Semantik

- Und wie implementiert man die?

- Wie lange hebt der Server Antworten an den Client auf? Bis Client neue Anfrage stellt.

BPI /OODS,
Prof. Hofmann
Oktober 1999

Bemerkungen zu Prüfung und Prüfer

- Angenehmes Klima; wenn man seine Fragen nicht sofort aus dem Kontext erkennt, gibt er zusätzliche Hilfen. Er verlangte kein Detailwissen. Die Zusammenhänge zwischen den einzelnen Kapiteln sind dagegen sehr wichtig. Benotung ist fair.

Fragen

- DCOM, CORBA, IDL

- Was sind die Proxies bei DCOM? (Es sind ganz normale Client-Stubs)

- Ist IDL eine Schnittmenge aller Zielsprachen? (Nein, es ist die Obermenge. Daher auch die Probleme beim Mapping)
- RPC (wie funktioniert das mit den Stubs?)
- Eignet sich Java für RPC's? (Nein, da Datenübergabe bei RPC per Value. Java kennt nur Referenzen)
- Aufrufsemantiken bei RPC? (mit vielen Transferfragen) (wieso sind starke Rechner bei exactly once wichtig? → stabiler Speicher; welche Aufrufsemantik bei Transaktionen? Wieso? → at-least-once; wie wird last-of-many transitiv? → Wiederholungs-ID)
- Wie können Orphans bekämpft werden? Welches Verfahren würden sie verwenden? (Kombination aus Extermination und Experation)
- Wieviele korrekte Knoten braucht ein verteiltes System? (n > 3m. zum Glück wollte er den Beweis nicht wissen :-))
- Was kann noch passieren, dass keine Einigung passiert? Und wieso? (asynchrone Kommunikation)

BP I, BP II, Hofmann Oktober 1999

Bemerkungen zu Prüfung und Prüfer

- Schwerpunktfach
- Scheinkolloquium
- Ergebnis: 1,0
- Angenehme Prüfungsatmosphäre, weniger Prüfung, mehr Gespräch. Er erzählt recht viel, ist immer sehr freundlich und nickt deutlich wenn ihm das gefällt, was man erzählt. Wenn man mal was falsch sagt oder nicht mehr weiterweiß dann hilft er durch weiteres nachfragen. Man sollte ihn ruhig ausreden lassen und nicht unterbrechen, wenn er was wissen will, dann fragt er schon. Gefragt ist hauptsächlich Verständniss und Zusammenhänge, manchmal auch ein bischen Detail. Aber wenn man sich bei einem Thema das er anspricht gut vorbereitet fühlt sollte man ruhig selber ein paar Details ansprechen, daß scheint ihm zu gefallen! Man sollte vielleicht NICHT BPI und II zur Prüfung nehmen, denn ich fand zwar die Vorlesung BPII sehr interesant und auf jeden Fall hörenswert, aber für die Prüfungsvorbereitung ist es allein von der Menge her nicht so gut. Besser wäre etwas kompakteres, z.B. Programmierung Paralleler Systeme. Wenn man in einigen Punkten aus dem Skript einfach nicht schlau wird kann man sich ruhig mal die Originalartikel ansehen. Zu mindest bei Speicherkonsistenzarten (BPII) war das extrem nützlich. Allerdings gibt's auch Artikel die so kompliziert sind, daß man auch nicht schlauer wird. Alles in allem eine sehr angenehme Prüfung, von der Note war ich sehr positiv überrascht.

Fragen

BPII

- Konsistenzprobleme bei Caches (Was ist das bzw. wie entstehen sie)
- Bsp. verteilte Dateiserver: was tut man dagegen (NIX! Wenn der Benutzer Konsistenz braucht, dann muss er sich halt drum kümmern)
- Speicherkonsistenzsemantiken (Sequentielle, Prozessor usw., alles recht oberflächlich erklärt). Kann man als normaler Mensch mit sequentieller Konsistenz gut umgehen? (jaja, ist das was man sich eigentlich immer vorstellt, wenn man programmiert)
- Markierung, Satz dazu
- Wird das mit der Freigabekonsistenz irgendwo implementiert (ich: keine Ahnung; wahrsch. nicht; Er: dochdoch, sogar in Hardware und zwar im soundso-Protokoll der sonstnochwo Uni und blablabla).
- Was kann man in HW sonst noch für Cachekons. tun? (MESI, grob erklärt, was welcher Zust. bedeuted und wie man hinkommt)
- Was macht man bei Mutliproz. zur Koord. was man bei Monopr. nicht macht? (Spinlocks. kurz erzählt warum, dann Ticket-lock

erklärt) Kann man dass dann einfach so in eine Bibliothek tun, z.B. JAVA und es fkt. überall? (Nein, braucht entsprechende atomare HW-Befehle(fetch-and-increment usw.) sonst wird's wieder ineffizient)

- Was noch? (Barrieren, auch da bischen erklärt)

BPI

- wie kann man verteilte atomare Aktionen machen (Transaktionen, den Weg von phys. System zur TA grob erklärt)
- Gibt's Verklemmungen? (Ja, wegen Seitensperren)
- Was tut man dagegen (Preclaiming der Seiten, oder Verkl.erkennung)
- Verklemmungserkennung, wie? (zentralisierter Algorithmus, Problem mit zusammenfassen der Nachr.; einfacher CMH)
- Auch wenn man Nachrichten nicht zusammenfasst kann's Probleme geben? (ja, weil sich Nachrichten überholen können, das wußte aber nicht ich, sondern ER).
- Verklemmung erkannt, was nun (Rücksetzen $\rightarrow$ Idempotente Aktionen nötig!)
- Rücksetzen sei kein Problem, Effizienz ist uns egal, was gibt es dann noch für Möglichkeiten eine Verkl. zu erkennen? (Wußte ich absolut nicht worauf er hinaus wollte. Nach ein paar weiteren Hilfsfragen hat ER aufgegeben, meinte „das ist wohl zu offensichtlich: Timeouts!").