Prüfungsfragen Betriebssysteme 2002

BP1 und OODS, Hofmann Oktober 2001

Bemerkungen zu Prüfung und Prüfer

- Prüfungsfach
- Ergebnis: 1,0
- Eher Gespräch als Prüfung.
- Versteht man eine Frage nicht, wird nachgefragt, bis man darauf kommt, was gemeint war.
- Man lernt auch in der Prüfung noch neue Zusammenhänge kennen.
- Details sind weniger wichtig, dafür aber um so mehr die Zusammenhänge (können auch über die Vorlesung hinaus gehen).

Fragen

- Was verbirgt sich hinter dem Begriff JINI?
- Welche Aufgabe hat der Lookup-Service bei JINI? → Auflösen von Service-Anforderungen.
- Was muß dieser liefern? → Referenz auf Stubs
- Umstieg auf Kommunikation: Wie funktioniert das Netzwerksystem im Betriebssystem? → Diagramm von Sende- und Emfangssystem von Folie 2-28 zeichnen und erklären
- Warum wird in realen Implementierungen oft der „netin"-Prozeß durch zwei Prozesse ersetzt, die durch eine Warteschlange verbunden sind? → Hardware könnte zu schnell Daten liefern und ein langsamer netin-Prozeß würde dafpr sorgen, daß die Hardware Pakete verwirft. Deswegen Aufteilung in schnellen netin- und langsameren ipin-Prozeß.
- Vergleichen Sie das ISO/OSI-Referenzmodell mit dem von TCP/IP. → Kann nicht 100%ig verglichen werden. Trotzdem: TCP/IP vierschichtig, Host2Network = Physical und Data Link Layer, IP = Network Layer, TCP (UDP) = Transport Layer. TCP/IP und UDP/IP besitzen auch noch Eigenschaften des Presentation Layer, weil sie eine bestimmte Byte-Order zur Kommunikation vorschreiben.
- Warum welche Nachteile hat ISO/OSI als Referenzmodell? → Häufiges Kopieren des Pakets bei jeder Schicht. Optimierungen: Verkettte Liste von Speicherblöcken bei neuen Netzwerkadaptern, oder Verletzung der Schichtenbildung durch Benutzung eines Offsets im Speicherbereich, der angibt, wieviele Bytes an Header noch vor das Paket müssen.
- Welchen Nachteil hat ISO/OSI noch? → Komposition von Protokollen nicht möglich. Wird gebraucht um Tunneling zu Realisieren. Beispiel: TCP/IP als „Physical Layer", IPX als Protokoll auf TCP/IP.
- Umstieg auf RPC und RMI: Was ist wünschenwert an RPC? → Transparenz für den Programmierer, gleiche Semantik für lokaler Prozeduraufruf (last-one). Erklärungen über die Wirklichkeit und Schwierigkeiten (keine vollst. Transp, da immer Anfrage beim Bindedienst).
- Was ist die Aufgabe von Stubs? → Marshalling, Request verschicken, Blockieren bis Antwort zurück, Wiederholungen usw. Eventuell Anfrage beim Bindedienst.
- Welchen Parameterübergabemechanismus verwendet JAVA? → Nur Call-by-Value!!! Objekte werden zwar als Referenz übergeben, damit läßt sich der Zustand des Objektes ändern, aber die Referenz selbst kann nicht umgebogen werden. Damit ist's kein rightiges Call-by-Reference.
- Wie funktioniert Marshalling bei JAVA RMI? → Objekt-Serialisierung.
- Welche Schwierigkeiten hat man noch bei RMI aus Sicht des Programmierers? → Aufruf-Sematik (exactly-once im fehlerfreien Fall, sonst don't know), man braucht deswegen noch höhere Konzepte in der Anwendung zur Fehlertoleranz.

Betriebsprogrammierung I u. II, Prof. Hofmann Oktober 2001

Bemerkungen zu Prüfung und Prüfer

- Ergebnis: 1,3
- Prüfer freundlich, wohlwollend
- Hilft, wenn man nicht gleich darauf kommt bzw. wenn man nicht weiß, worauf er hinaus will
- stellt auch Transferfragen
- Hat ein Thema für die Prüfung (bei mir "gegenseitiger Ausschluß")
- Benotung: fair

Fragen

Betriebsprogrammierung II

- Gegenseitiger Ausschluss bei Multiprozessoren:
- Wie? (Spinlocks)
- Bei welchen Anwendungen sinnvoll? (Bei Programmen mit einem Prozess pro Prozessor - hab ich nicht gewusst, floss aber offensichtlich nicht in die Bewertung ein (?!))
- Algoritmen? (Alle erklaert: test_and_set, test_and_set mit exponentiellem Warten, test_and_test_and_set, ticket-lock, array-based lock, list-based lock: alle Algoritmen erklaert, jeweils mit Vor- und Nachteilen: Speicheraufwand, Cache-Invalidierungen,...)
- Verteilter gemeinsamer Speicher (=DSM) bei NORMA-Architekturen: (Strategien migrierend, nichtmigr., replizierend, nichtreplizierend kurz erklaert)
- Dann die Transferfrage: Welche der Spinlock Algorithmen mit welchen DSM-Strategie (kam ich nicht gleich drauf, hat er etwas geholfen: list-based am besten bei migrierend und lese-replizierend (?), array-based und test_and_test_and_set bei nur replizierend (?), test_and_set und ticket-lock bei zentralisiert (?); Analogie der Strategien mit kohärenten/nicht kohär. Caches - das hat ER gewusst...)

Betriebsprogrammierung I

- gegenseitiger Ausschluss _ohne_ gemeinsamen Speicher? (Erlaubnis- und Tokenbasierte Algorithmen)
- Beschreiben Sie einige! (Lamport genau beschrieben, Ricart/Argrawala ebenso und beide verglichen, dabei natürlich logische Zeit erwähnt)
- Logische Zeit erklären! (Bildchen hingemalt und kurz Regeln für Uhrfortschritt bei den unterschiedl. Ereignissen erklärt, dann bin ich auf die _einseitige_ Implikation für kausale Abhängigkeit gekommen, deshalb nächste Frage)
- Gibt es auch Modifikation, bei denen kausale Abhängigkeiten eindeutig sind? (Vektoruhren, komponentenweiser Größenvergleich, falls nicht vergleichbar: Ereignisse nebenläufig)
- Allgemeiner, erlaubnisbasierter Algorithmus (Stichwort: Sanders, Request- und Informationsmengen erklärt, formale Voraussetzungen an diese, Algorithmus in groben Zügen erklärt, nicht deadlockfrei, Deadlockbehandlung wie bei Maekawa (mit INQUIRE und RELINQUISH) nur gesagt, nicht erklärt)
- Algorithmus von Singhal erklären! (Dreiecksstruktur der Matrix", nur Requests an Prozesse, die das Token haben können - und dann war die Zeit schon um...)