Aufgabe K17: Kanal mit Durchschlag

Ein spezieller Kernobjekttyp „Durchschlagskanal“ stellt Nachrichten zweimal zu: erst als Original an einen Empfänger, dann als Durchschlag an einen Auditor. Der Auditor erhält zusätzlich die Kennungen von Sender und Empfänger. Um Manipulationen des Durchschlags zu verhindern, wird jede Nachricht zuerst in einen beschränkten Kanalpuffer kopiert. Ist der Puffer voll, scheitert das Senden. Dieser Kanaltyp bietet folgende Operationen an:

`sendAsyn`: sendet eine Nachricht, ohne zu warten

`receiveSyn`: empfängt eine Nachricht im Original, wartet bis zum Eintreffen

`receiveAuditSyn`: empfängt eine Nachricht als Durchschlag, wartet bis zum Eintreffen

Die Implementierung verwendet folgende Datenstrukturen, initial sind alle leer:

P: Pufferspeicher für Nachrichteninhalte

MO: Menge für Originale, also neue Nachrichten

MD: Menge für Durchschläge, also zugestellte Nachrichten

WE: Wartemenge für Empfänger von Originalen

WA: Wartemenge für Auditoren, also Empfänger von Durchschlägen

a) Welche Informationen speichern die Elemente der vier Mengen?
Hinweis: Die Antwort ergibt sich aus den Ablaufbeschreibungen.

MO: (Nachrichtenreferenz, Senderkennung)

MD: (Nachrichtenreferenz, Senderkennung, Empfängerkennung)

WE: (Empfangspufferreferenz, Empfängerprozess)

WA: (Auditpufferreferenz, Auditorprozess)

Nachrichten durchlaufen zwei Stationen, repräsentiert durch MO und MD. Auditoren erwarten sowohl Sender- als auch Empfängerkennung. In MO ist nur der Sender bekannt, in MD auch der Empfänger.

WE und WA sind die Wartemengen für blockierte Empfänger bzw. Auditoren. Sie verwenden laut den Teilaufgaben unterschiedliche Arten von Puffern. Ein Empfangspuffer kann nur die Nachricht aufnehmen, ein Auditpuffer zusätzlich die Kennungen von Sender und Empfänger. Man könnte für Auditoren auch gewöhnliche Empfangspuffer nehmen und zusätzliche Argumente für die Kennungen vorsehen.

Die Aufgabenstellung beschreibt nicht, welche Form die Kennungen von Sender und Empfänger haben. Eine Prozesskennung wäre in der Praxis für ein Audit–Log unbrauchbar, da sie sich im Nachhinein keiner Instanz und keinem Programm mehr zuordnen lässt. Das ist aber auch nicht Thema der Aufgabe.

b) Beschreiben Sie den Ablauf des Durchschlagempfangens. (`receiveAuditSyn`)
Argument ist die Referenz auf einen Auditpuffer.

Die einfachste der drei Operationen. Der Auditor erwartet ein Element in MD.

- MD leer? +1

ja:
- Aufrufer blockieren +0.3
- (Auditpufferreferenz, Aufrufer) in WA legen +0.7

nein: *abholen...*

- (Nachrichtenref., Sender-, Empfängerkennung) aus MD nehmen +1
- Nachricht aus P in Auditpuffer kopieren +1
- Sender- und Empfängerkennung in Auditpuffer kopieren +1
- Nachricht aus P löschen — *nicht vergessen!* +1

c) Beschreiben Sie den Ablauf des synchronen Empfangens. (`receiveSyn`)
Argument ist die Referenz auf einen Empfangspuffer.

Diese Operation kann zwei Ereignisse auslösen, nämlich das Zustellen an den aufrufenden Empfänger und das Zustellen des Durchschlags an einen Auditor. Der Empfänger erwartet ein Element in MO.

- MO leer? +1

ja:
- Aufrufer blockieren +0.3
- (Empfangspufferreferenz, Aufrufer) in WE legen +0.7

nein: *abholen...*

- (Nachrichtenreferenz, Senderkennung) aus MO nehmen +0.5
- Nachricht in Empfangspuffer kopieren +1
- Kennung des Empfängers, also des Aufrufers, bestimmen +0.5
- WA leer? +1

ja: *kein Auditor, Durchschlag muss warten*
- (Nachrichtenref., Sender-, Empfängerkennung) in MD legen +1
- Ende des Aufrufs

nein: *zustellen...*

- (Auditpuffer, Auditor) aus WA nehmen +0.5
- Nachricht in Auditpuffer kopieren +1
- Sender- und Empfängerkennung in Auditpuffer kopieren +1
- Auditor deblockieren +0.5
- Nachricht aus P löschen — *nicht vergessen!* +1

d) Beschreiben Sie den Ablauf des asynchronen Sendens. (`sendAsyn`)
Argument ist die Referenz auf eine Nachricht im Adressraum des Senders.

Auch diese Operation kann zwei Ereignisse auslösen. Das Zustellen eines Durchschlags an den Auditor verläuft exakt wie beim entsprechenden Fall in `receiveSyn`.

Laut Aufgabenstellung wird die Nachricht auf jeden Fall in einen Kanalpuffer kopiert, auch wenn ein Empfänger bereitsteht. Das verhindert, dass weitere Prozesse im Adressraum von Sender oder Empfänger die Nachricht verändern, nachdem sie vom Kanal abgeholt wurde, aber bevor der Auditor einen Durchschlag erhält. Natürlich muss man dazu auch immer die Kopie aus dem Kanalpuffer übergeben.

- Platz für Nachricht in P? +0.5

nein: Abbruch mit Fehler +0.5

- Nachricht in P kopieren, neue Referenz NPRef auf Nachricht in P +1
- Kennung des Senders, also des Aufrufers, bestimmen +0.5

- WE leer? +1

ja: *kein Empfänger, Nachricht muss warten*

  – (NPRef, Senderkennung) in MO legen +1
  – Ende des Aufrufs

nein: *zustellen...*

- (Empfangspuffer, Empfänger) aus WE nehmen +0.7
- Kennung des Empfängers bestimmen +0.5
- Nachricht aus P in Empfangspuffer kopieren +1
  wichtig: aus P, nicht aus dem Adressraum des Senders
- Empfänger deblockieren +0.3

- WA leer? +1

ja: *kein Auditor, Durchschlag muss warten*

  – (NPRef, Senderkennung, Empfängerkennung) in MD legen +1
  – Ende des Aufrufs

nein: *zustellen...*

- (Auditpuffer, Auditor) aus WA nehmen +0.5
- Nachricht aus P in Auditpuffer kopieren +1
  wichtig: aus P, nicht aus dem Adressraum des Senders oder Empfängers
- Sender- und Empfängerkennung in Auditpuffer kopieren +1
- Auditor deblockieren +0.5
- Nachricht aus P löschen — *nicht vergessen!* +1