

# Lösung von Synchronisationsaufgaben mit Semaphoren

Prof. Dr. Carsten Vogt, FH Köln, Institut für Nachrichtentechnik, [www.nt.fh-koeln.de/vogt/](http://www.nt.fh-koeln.de/vogt/)

Ein Schwerpunktthema der Vorlesung "Betriebssysteme" / "Verteilte Systeme" ist die Synchronisation nebenläufiger Prozesse oder Threads. Es geht dabei darum, ein bestimmtes Zeitverhalten der Prozesse durchzusetzen, also zum Beispiel einen wechselseitigen Ausschluss oder eine Reihenfolge. Solche Aufgaben spielen in der Praxis eine große Rolle: Will man beispielsweise eine Datenbank korrekt implementieren, so muss man dafür sorgen, dass ein Datenbereich jeweils nur von einem Prozess überschrieben werden kann, da sich mehrere Schreiber gegenseitig stören würden. Ein wichtiges Werkzeug zur Prozess-Synchronisation sind Semaphore.

Damit die Aufgabenstellungen nicht so trocken sind, stelle ich sie oft anschaulich durch Situationen aus dem täglichen Leben dar. Dahinter stehen jedoch immer Synchronisationsprobleme, die auch bei der Realisierung eines nebenläufigen Programms auftreten können. Ich möchte Ihnen hier eine kurze Anleitung geben, wie man solche Probleme systematisch lösen kann.

## Beispielaufgabe

Ein Warenautomat enthält zu Beginn 100 Blumensträuße. Er führt dann immer wieder die folgenden Schritte aus: Ist er leer, so wartet er, bis der Lieferant kommt (siehe unten). Anschließend wartet er, bis ein Kunde fünf Euro eingeworfen hat, und gibt dann einen Strauß aus.

Es gibt mehrere Kunden. Jeder von ihnen führt die folgenden Schritte aus: Benutzt gerade ein anderer Kunde den Automaten, so wartet er, bis der Automat wieder frei ist. Er wirft dann fünf einzelne Eurostücke ein, wartet, bis die Ware kommt (was durchaus länger dauern kann, wenn der Automat gerade leer ist) und gibt den Automaten wieder frei.

Ein Lieferant kommt alle zwei Tage und füllt 100 Blumensträuße nach (unabhängig davon, wie viele Sträuße noch im Automaten vorhanden sind).

## Schritt 1: Identifizieren Sie die Prozesse des Systems

Die Prozesse sind die Aktivitäten, die nebenläufig ausgeführt werden und sich dabei synchronisieren müssen. Sie ergeben sich meist unmittelbar aus der Problemstellung.

*In der Beispielaufgabe sind Automat, Kunden und Lieferant die Prozesse.*

## Schritt 2: Skizzieren Sie die Ausführungsschritte der einzelnen Prozesse

Notieren Sie in Stichworten die Aktionen der einzelnen Prozesse. Dabei muss deutlich werden, welcher Prozess was tut, in welcher zeitlichen Reihenfolge die Aktionen eines Prozesses ablaufen und wo es Schleifen oder Verzweigungen gibt. Die Aktionen eines Prozesses und ihre Abfolge ergeben sich meist unmittelbar aus der Aufgabenbeschreibung.

Am besten zeichnen Sie für jeden Prozess eine eigene Spalte, in die Sie die Aktionen des Prozesses eintragen. Synchronisationsoperationen (also insbesondere die P- und V-Operationen auf Semaphoren) müssen Sie hier noch nicht einfügen; sie werden in Schritt 5 ergänzt.

*Die Prozesse der Beispielaufgabe laufen wie folgt ab:*

<i>Automat</i>	<i>Kunde</i>	<i>Lieferant</i>
<i>while (true) {   warte auf Liefer., falls leer;   warte auf 5 Ein-Euro-Stücke;   gib Blumenstrauß aus; } </i>	<i>warte, bis Automat frei;   wirf 5 Ein-Euro-Stücke ein;   warte auf Blumenstrauß;   gib Automaten frei; </i>	<i>while (true) {   warte zwei Tage;   fülle 100 Sträuße nach; } </i>

### Schritt 3: Identifizieren Sie die Synchronisationsbedingungen

Eine Synchronisationsbedingung ist eine zeitliche Beziehung, die zwischen zwei oder mehr Prozessen durchgesetzt werden muss. Die Synchronisationsbedingung selbst ist also kein Prozess, sie ist vielmehr eine Abhängigkeit zwischen Prozessen. Man identifiziert die Bedingungen am besten dadurch, dass man die Problembeschreibung genau durchliest und jede zeitliche Bedingung, die dort beschrieben wird, in einer Liste notiert oder mit einer eigenen Farbe markiert.

*In der Beispielaufgabe gibt es vier Synchronisationsbedingungen:*

- a.) Nur ein Kunde gleichzeitig benutzt den Automaten. (wechselseitiger Ausschluss)
- b.) Ein Kunde wartet, bis der Automat die Ware liefert. (einfache Reihenfolge)
- c.) Der Automat wartet, bis ein Kunde 5 Ein-Euro-Stücke eingeworfen hat. (erweiterte Reihenfolge)
- d.) Der leere Automat wartet, bis der Lieferant 100 Sträuße gebracht hat. (erweiterte Reihenfolge)

### Schritt 4: Definieren Sie die Semaphore

Jede der Synchronisationsbedingungen muss mit einem eigenen Semaphor durchgesetzt werden. Geben Sie jeweils an, wie der Semaphor heißen soll, welcher Bedingung er zugeordnet ist und wie er initialisiert wird (das heißt, welchen Anfangswert er bekommt). In vielen Fällen liegt der Anfangswert unmittelbar auf der Hand – bei einem wechselseitigen Ausschluss ist es 1, bei einer einfachen Reihenfolgebeziehung ist es 0.

*Die Semaphore der Beispielaufgabe und ihre Initialisierungen sehen wie folgt aus:*

- a.) *WA.INIT(1): zum wechselseitigen Ausschluss der Kunden.*
- b.) *WARE.INIT(0): zur Blockierung des Kunden bis zur Warenübergabe.*
- c.) *GELD.INIT(0): zur Blockierung des Automaten bis zur Geldübergabe.*
- d.) *FÜLLSTAND.INIT(100): zur Blockierung des Automaten, falls er leer ist.*

### Schritt 5: Ergänzen Sie die "Programme" aus Schritt 2 durch Semaphoreoperationen

Nun müssen Sie nur noch die P- und V-Operationen der Semaphore, die in Schritt 4 eingeführt wurden, in die Programme aus Schritt 3 einfügen. P-Operationen müssen dabei an die Stellen gesetzt werden, an denen ein Prozess blockiert wird, bis ein bestimmtes Ereignis eintritt – der Prozess also auf etwas warten muss. V-Operationen kommen an die Stellen, an denen ein Prozess einen anderen entblockiert – ihm also mitteilt, dass er jetzt weiterlaufen kann.

*Für das Beispielproblem lauten die vollständigen Programme folgendermaßen:*

<i>Automat</i>	<i>Kunde</i>	<i>Lieferant</i>
<pre>while (true) {   FÜLLSTAND.P(1);   (= senke Anzahl Blumen u.    warte auf Lief., falls leer)   GELD.P(5);   (= warte auf 5 1-€-Stücke)   WARE.V(1);   (= gib Blumenstrauß aus,    entblock. damit Kunden) }</pre>	<pre>WA.P(1); (= warte, bis Automat frei) for (i=0;i&lt;5;i++)   GELD.V(1); (= wirf 5 1-€-Stücke ein) WARE.P(1); (= warte auf Blumenstrauß) WA.V(1); (= gib Automaten frei)</pre>	<pre>while (true) {   warte zwei Tage;   FÜLLSTAND.V(100);   (= fülle 100 Sträuße nach) }</pre>

**Anmerkung zu Prüfungen:** In einer Klausur müssen Sie nur die Ergebnisse der Schritte 4 und 5 hinschreiben – es sei denn, es wird ausdrücklich anders verlangt.