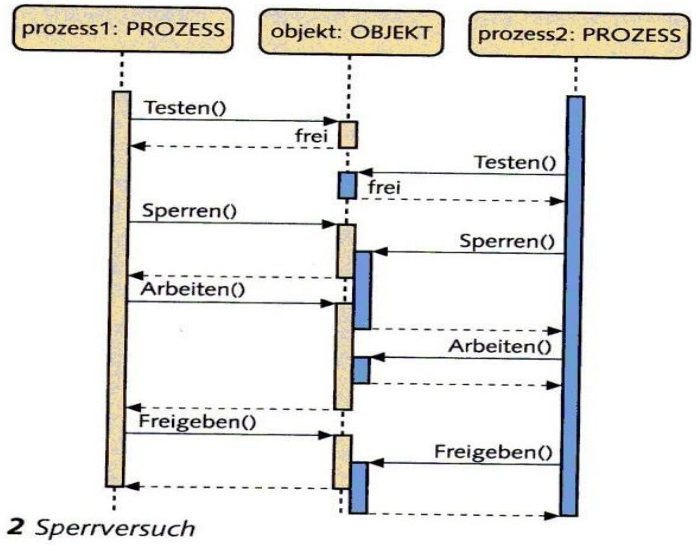
Modellierung einfacher nebenläufiger Prozesse – mögliche „Verklemmung“

(vgl. Buch [Oberstufe 2] S.77ff.)

Immer dann wenn zwei oder mehr Prozesse **nebeneinanderlaufen (parallel)** und die gleichen Ressourcen benutzen (Dateien, Drucker, usw.), sei es innerhalb eines Programms oder z.B. bei Server-Client-Anwendungen besteht die Gefahr von Fehlern (z.B. wenn ein Prozess eine Variable ändert, die ein anderer gerade verändert). **Verklemmungen** (Deadlock) können immer dann auftreten wenn nebenläufige Prozesse mehr als eine gemeinsame Ressource benötigen und sich damit blockieren(vgl Buch S.81 Abb.6). Um solche Probleme darzustellen und zu identifizieren eignen sich so genannte **Sequenzdiagramme.**

Auf der y-Achse wird der Verlauf der Prozesse dargestellt, die „Balken“ in der Mitte unter dem da dargestellten Objekt geben an, welcher der Prozesse gerade Rechenzeit erhalten hat (der andere Prozess „ruht“ also). Links und rechts davon werden die ausgeführten Arbeitsschritte gezeigt (z.B. Testen() oder Arbeiten()):



Die Abbildung zeigt zugleich einen gescheiterten Versuch das Problem der Nebenläufigkeit zu beheben und ein Beispiel eines Sequenzdiagramms. Die Idee ist die Prozesse vorm Arbeiten das Objekt auf Verfügbarkeit zu überprüfen, also zu testen ob bereits ein anderer Prozess darauf zugreift. Das Problem: Wenn vor dem Sperren() ein anderer Prozess die Verfügbarkeit testet, arbeiten und sperren als Folge beide Prozesse unabhängig voneinander das Objekt → Nebenläufigkeit. Es wird klar das dieses Problem nicht auf Programmebene gelöst werden kann. Es müssen spezielle Maschinenbefehle in die Prozessoren eingebaut werden und das Betriebssystem muss Methoden zur verfügung stellen um diese zu benutzen. Eine prüft und sperrt gegebenfalls, eine zweite gibt ein gesperrtes Objekt wieder frei. Das Prinzip dieser „Nutzungsmethoden“ nennt man **Semaphorenprinzip.**