

1)

14.13

Zunächst wird der Betrag ausgewählt, den auswählen möchte.
wurde der Betrag ausgewählt, kommt man zum "Ende",
wenn man einen anderen Betrag auswählen möchte,
kommt man zu "einer Anant". bestätigt man jetzt den
Betrag, kommt man zum Ende
wird während dem Prozess irgendwo abgebrochen,
so kommt man zum Kettensprung.

14.32)

Idle ist der Startpunkt.

der nächste Schritt ist Req(1d), wird dieser Schritt durchgeführt
kommen wir zu einer Verzweigung.

ist die $1d > 10$, so kommen wir zu Major(1d) und ausschließlich
zu MajorReq := 1d; und dann zu zur Kreuzung.

ist die $1d \leq 10$, so kommen wir zu Minor(1d) und ausschließlich
zu MinorReq := 1d; und dann zur Kreuzung.

nach der Kreuzung endet das ganze bei "Busy".

1)

14,7)

Knoten

Anruf wird mit dem "Start des Anruftours begonnen

Anruf wird mit dem "Stoppen des Anruftours beendet

Wird die Nummer eingegeben, so gelien wir zum Zustand "Partial Dial" zu. Es werden so viele Ziffern eingegeben, bis eine valide Rufnummer auf dem Telefon ausgewählt wurde. Ist die Rufnummer valide, so gelien wir zum Ausstiegspunkt.

14,14)

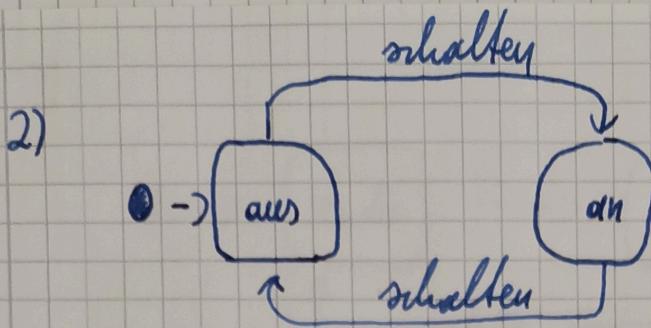
Der Baukastenautomat verifiziert zunächst die Karte.

wird die Karte abgelehnt, gelien wir zum nächsten Zustand "read amount : Readamount 5.11"

wird die Transaktion unterbrochen, so gelien wir zu "Cardrelease".

wird die Transaktion nicht unterbrochen, so gelien wir zu "verify Transaction". ~~wird die Karte ausliefernd~~ geliefert es zu Zustand "Cardrelease"

Funktioniert der Automat nicht, gelien wir zu Zustand "out Of Service"

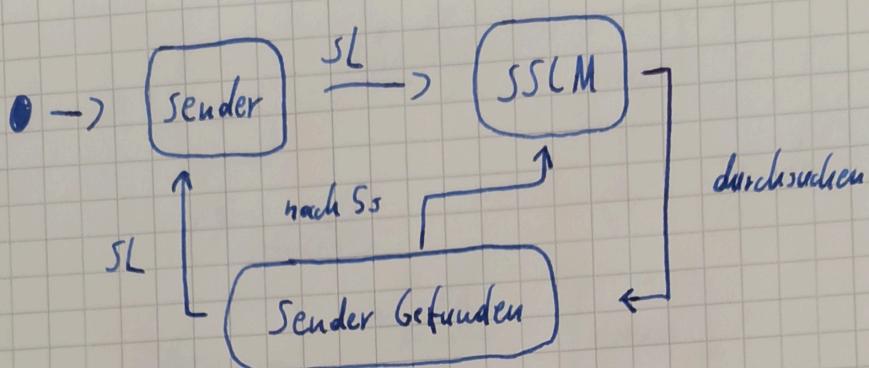


```
#include <iostream>
bool switching
```

```
int main(){
    int x=1
    char s;
    const char *t = "Y";
    for(bool i=false; x>0; x++){
        std::cout << "do you want to turn the switch? [Y/N]" << std::endl;
        std::cin >> s;
        if((*t)==s){
            i=switching(i);
        }else{
            break;
        }
        std::cout << i << std::endl;
    }
    return 0;
}
```

```
bool switching(bool i){
    if(i==false){
        i=true;
    }else{
        i=false;
    }
    return i;
}
```

3)



4) Quelle UML26Lasklar Rupp-Chris S.370

Die UML definiert Pseudozustände, um komplexe Beziehungen zwischen Zuständen einfach darzustellen. Dies wird dadurch realisiert, dass Transitionen nicht nur zwischen Zuständen modelliert werden, sondern auch über einen oder mehrere Pseudozustände gelten dürfen.



5)

