1. Sequenz-Diagramm → Quellcode Implementieren Sie die Klasse Nikohaus passend zu den folgenden Vorgaben und zeigen Sie die Klasse in einem Fenster an, wo Sie per Button Nikohaus.drawIt ein- oder ausschalten:

```
import javax.swing.*;
import java.awt.*;
public class Nikohaus extends JPanel {
   private boolean drawIt = false;
   private int x = 100;
   private int y = 100;
   private int w = 100;
   private int h = 100;
   private void draw(Graphics g){
      try {
         if (drawIt) {
            g.drawLine(x, y + h, x, y + h / 2);
            Thread.sleep(500L);
            g.drawLine(x, y + h / 2, x + w / 2, y);
            Thread.sleep(500L);
            g.drawLine(x + w / 2, y, x + w, y + h / 2);
            Thread.sleep(500L);
            g.drawLine(x + w, y + h / 2, x, y + h / 2);
            Thread.sleep(500L);
            g.drawLine(x, y + h / 2, x + w, y + h);
            Thread.sleep(500L);
            g.drawLine(x + w, y + h, x, y + h);
            Thread.sleep(500L);
            g.drawLine(x, y + h, x + w, y + h / 2);
            Thread.sleep(500L);
            g.drawLine(x + w, y + h / 2, x + w, y + h);
         }
      }catch(InterruptedException e){
      }
   }
   public void setDrawIt(boolean b){
      drawIt = b;
      this.repaint();
   }
   @Override
   protected void paintComponent(Graphics g) {
      super.paintComponent(g);
      draw(g);
   }
   public boolean isDrawIt(){
      return drawIt;
   }
}
```

4. Recherche

Finden Sie mit Hilfe eines guten UML-Buchs die Antworten auf folgende Fragen. Bringen Sie dieses UML-Buch mit zum Praktikum (bei einem E-Buch genügt das PDF) und zeigen Sie die Stellen mit den entsprechenden Erklärungen.

(a) . Wie können die folgenden Konzepte in Sequenz-Diagrammen dargestellt werden?

i. Verzweigung

ii. Schleife

iii. Schachteln von Sequenz-Diagrammen ineinander

iv. lokale Variablen/Attribute

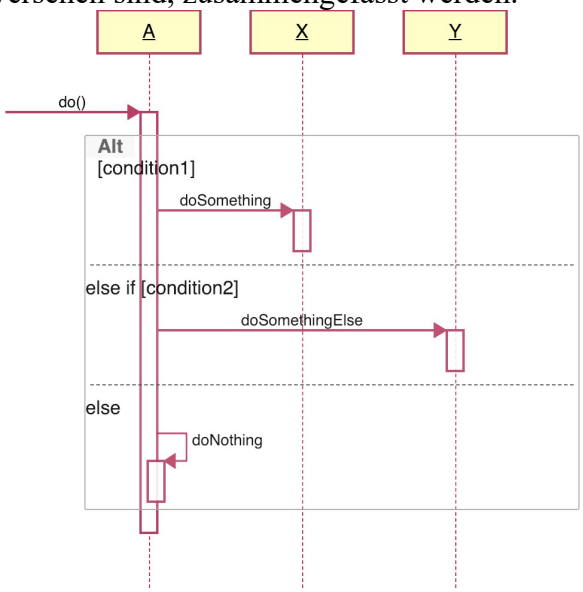
v. Kommunikationspartner ist aktiv/nicht aktiv, d.h. befindet sich/befindet sich nicht in einer „Ausführungssequenz“ („execution specification“)

(b) Nennen Sie mindestens fünf Beispiele für Modellelemente/Konzepte, die in Sequenz- aber nicht in Kommunikations-Diagrammen dargestellt werden können.

(c) UML-Diagramme können in einen Rahmen eingezeichnet werden, in dessen linker oberer Ecke u.a. die Art des Diagramms angegeben wird. Wie sehen diese Diagrammrahmen im Allgemeinen aus? Welche Abkürzungen stehen in der linken oberen Ecke des Rahmens für welche Diagrammart?

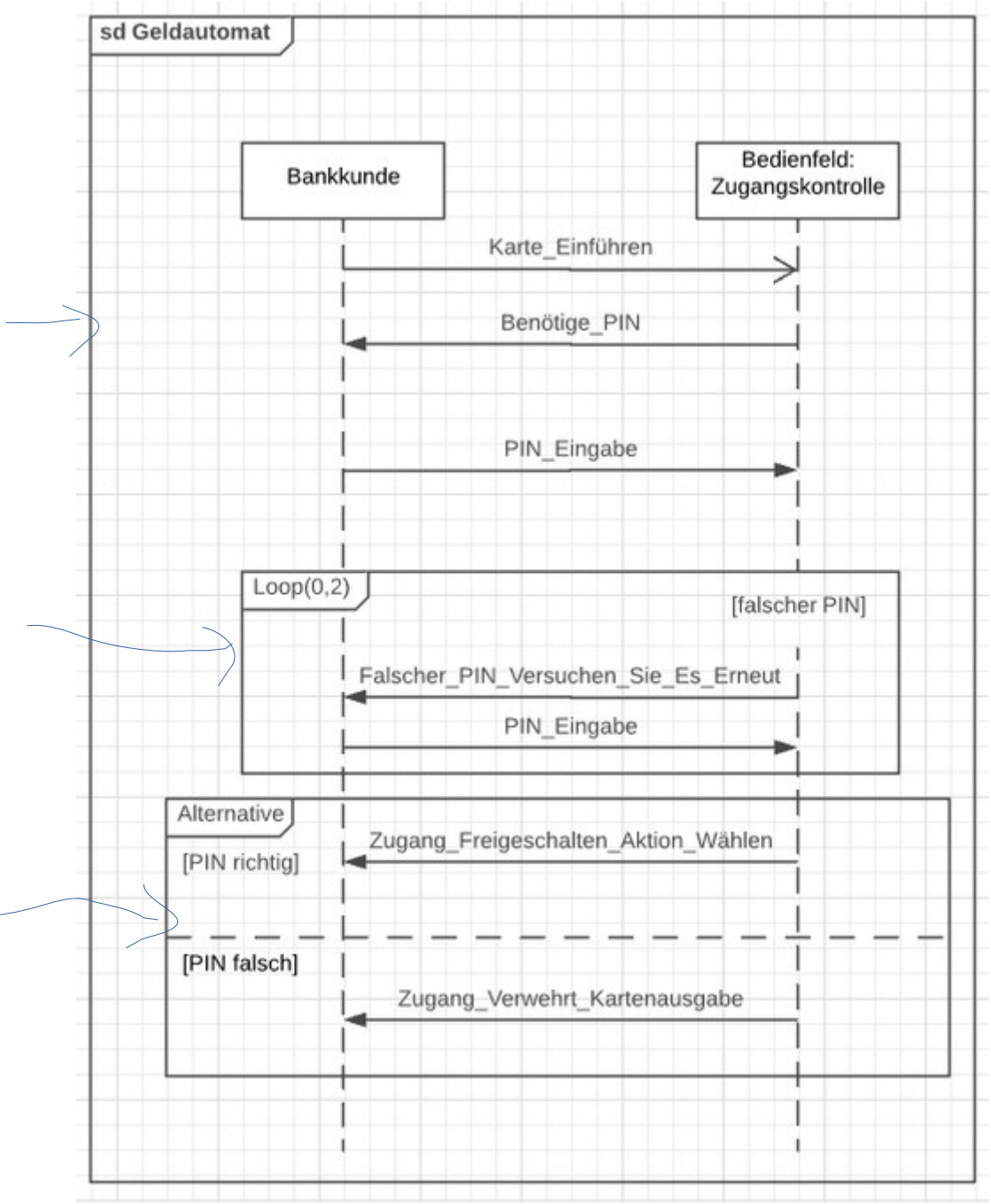
a)

Verzweigung: Durch einen alt-Operator können alternative Abläufe, die durch Bedingungen versehen sind, zusammengefasst werden.

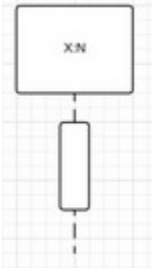


Schleife: Mit Hilfe des loop-Operators können Schleifen definiert werden. Zur Vereinfachung findet man manchmal auch *loop while* oder *loop until*.

Schachteln von Sequenzdiagrammen: In UML können Sequenzdiagramme erstellt werden, die verschachtelte Teilsegmente beinhalten. Rahmen helfen, die einzelnen Fragmente geordnet darzustellen.



Lokale Variablen/Attribute: Operanden werden mit Einem Rechteckigen Kopf und einer gestrichelten Lebenslinie dargestellt, im Kopf steht der Objekt und Klassenname.



Kommunikationspartner aktiv/inaktiv: Zur Bearbeitung eines Methodenaufrufs ist ein Objekt eine gewisse Zeit aktiv. Zur Darstellung dieser Aktivität dient der Aktivitätsbalken, der bei Rekursion auch geschachtelt auftreten kann.

b)

TABELLE 16.1 „Fehlende" Modellkonstrukte in Kommunikationsdiagrammen

Modellierungselement	Darstellung	Kapitelreferenz
Attribute von Lebenslinien	Kommunikationspartner.Attribut=Wert	15.4.2
Aktion und Aktionssequenz		15.4.2
Destruktion von Lebenslinien	lösche	15.4.2
Zustandsinvariante	{Bedingung} Bedingung	15.4.4
Kombiniertes Fragment	operator	15.4.5
Ordnungsbeziehung		15.4.6
Interaktionsreferenz	ref	15.4.7
Verknüpfungspunkt (Gate)	Gatename	15.4.8
Zerlegung von Lebenslinien	Name des Kommunikationspartners **ref** Name der referenzierten Interaktion	15.4.9

c)

Standardnotation

Es mag verwundern, aber die UML 2 sieht für Diagramme *kein explizites Notationselement* vor. In aller Regel wird in der Praxis diese Lücke werkzeugabhängig gelöst. Sehr häufig besteht ein Diagramm aus einem leeren Zeichenblatt, dem bei seiner Erstellung ein Name und Typ zugewiesen wird.

Optional und für bestimmte Fälle sieht die UML jedoch einen Rahmen (Frame) vor, der bestimmte Modellelemente umschließt (Abbildung 3.2). Dabei muss mindestens ein Name für die dargestellten Inhalte im Rahmenkopf (Heading), einem Fünfeck im Rahmen, stehen, daneben können optional der Typ und zusätzliche Parameter eingetragen werden.

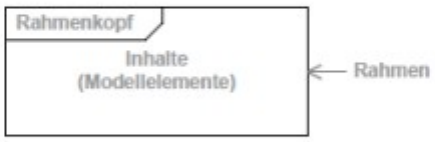


ABBILDUNG 3.2 Notation eines Diagrammrahmens

Die abstrakte Notation des Rahmenkopfs lautet:

<Rahmenkopf> ::= [<**Typ**> | <**Kürzel**>] <Name> [<(Parameter)>]

Dabei können Sie den Typ bzw. alternativ das Kürzel der Tabelle 3.2 entnehmen.

TABELLE 3.2 Rahmentypen und deren Kürzel

Typ	Kürzel
class	–
package	pkg
component	cmp
deployment	dep
use case	uc
activity	act
state machine	stm
interaction	sd

Um die gesamte Bezeichnung im Rahmenkopf zu verstehen, müssen wir zunächst die Fälle erläutern, in denen der Rahmen eingesetzt wird. Die UML 2 erlaubt den Aufbau von hierarchischen Modellstrukturen, das Auslagern und die Mehrfacheinbindung von Teilen eines Diagramms sowie die Bildung von Namensräumen (z. B. durch ein Paket) und Besitzstrukturen zwischen Elementen (z. B. eine Klasse besitzt ein Verhalten modelliert als Zustandsautomat).

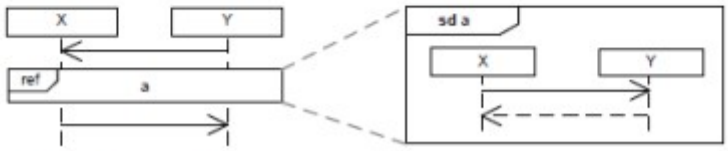
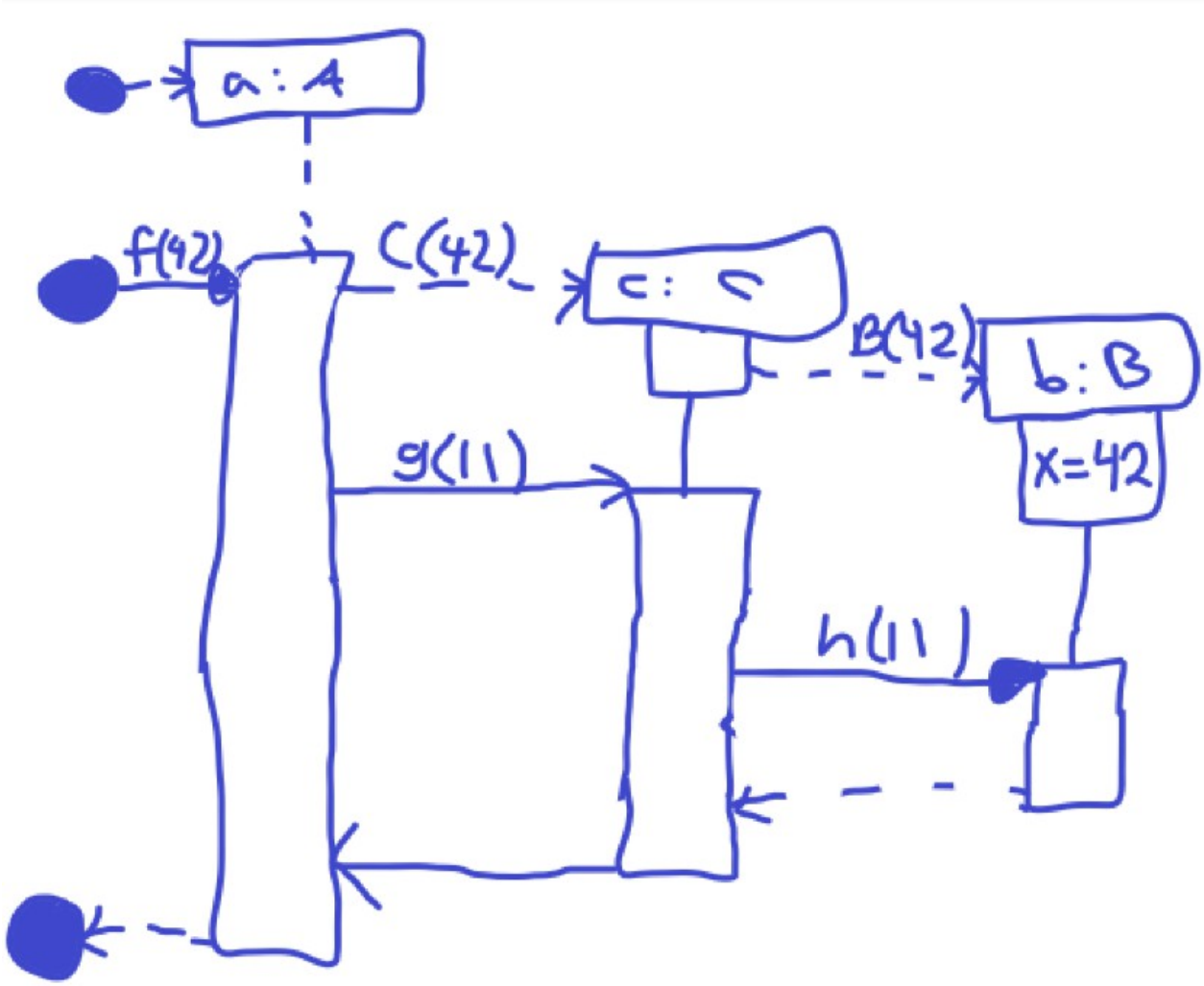


ABBILDUNG 3.3 Auslagern von Diagrammteilen

2)



3)