Objektorientierte Modellierung am Beispiel von Pacman

Bei der Objektorientierten Modellierung stellt sich das Problem einen Vorgang mit Hilfe von Objekten zu beschreiben.. Dies soll an dem Beispiel des Spiel **PacMan** (Online-Version: **http://www.neave.com/games/**) erläutert werden.

Am Anfang steht eine Beschreibung des Spiels in Worten. Ein Beispiel dafür könnte der folgende Text sein.

- Pacman lässt sich durch die Tastatur steuern.
- Er bewegt sich nur, wenn es im Labyrinth möglich ist.
- Trifft er dabei auf einen Punkt, so erhöht sich sein Punktestand um 1.
- Trifft er auf einen dicken Punkt, so erhöht sich sein Punktestand um 10 Punkte. Außerdem wird er für 20 Takte unverwundbar.
- Trifft er auf einen Geist und ist verwundbar, so wird er gefressen und er hat ein Leben weniger. Ist er jedoch unverwundbar, so stirbt der Geist und der Punktestand erhöht sich um 100 Punkte.
- PacMan bewegt sich gleich schnell wie die Geister
- Die Geister verfolgen PacMan, solange er verwundbar ist und fliehen vor ihm, wenn er unverwundbar ist.
- Die Geister können sich nur innerhalb des Labyrinth bewegen.
- Das Spiel endet, wenn alle Punkte eingesammelt sind oder PacMan kein Leben mehr hat.
- Die erreichte Punktezahl wird ausgegeben.
- Am Ende des Spiels wird der Gewinner angegeben.

Aus diesem Text werden nun die Objekte ermittelt. Eine einfache Faustregel besagt, dass man zu jedem Substantiv eine Klasse entwirft und zu jedem Verb ein Methode. Dies lässt sich jedoch so generell nicht durchziehen. Manche Substantive sind auch Attribute von Objekten.
Allgemein gilt:

Objektorientierte Modellierung
In einem ersten Schritt wird für jedes Substantiv eine Objekt entworfen. Dann wird die Struktur genauer untersucht. Manche Objekte beschreiben das gleiche und lassen sich in eiern Klasse zusammenfassen. Manche Objekte lassen sich besser als Attribute formulieren. Nach der Festlegung der Objekte bestimmt man die Attribute. Hier liefern die Eigenschaften in der Beschreibung wichtige Hinweise. Die Verben in der Beschreibung führen zu den Methoden. Zum Schluss werden gemeinsame Eigenschaften in einer Hierarchie zusammengefasst.

Für das obere Problem ergeben sich damit als mögliche Objekte:

Pacman	**Tastatur**	**Labyrinth**	**Punkte**	**Punktestand**
Takte	**Geist**	**Spiel**	**Leben**	

Eine genauere Betrachtung ergibt jedoch:
– Die Punkte sind eine Eigenschaft des Labyrinths.
– Der Punktestand gehört zum Objekt Spiel
– Die Takte sind nur für PacMan wichtig und geben an wie lange er unverwundbar ist. Daher scheiden diese als Objekte aus.
– Leben kann als Attribut von PacMan verwaltet werden, da es jedoch auch über das Spielende entscheidet, wird es als Attribut dem Objekt Spiel zugeordnet.
– Ende wird als Methode des Objekts Spiel realisiert.

Damit bleiben noch 5 Objekte über. Für diese Objekte müssen nun die Attribute und Methoden festgelegt werden.

Pacman	**Tastatur**	**Labyrinth**	**Geist**	**Spiel**

Beginnen wir in der obigen Reihenfolge zuerst mit den Attributen:

PacMan	Tastatur	Labyrinth	Spiel	Geist
x, y Form Uhr Farbe unver- wundbar	Taste gedrueckt	Plan Punkteanzahl	Punktestand Leben	x, y Form Farbe lebendig Punkte

Die Uhr gibt an, wie lange PacMan noch unverwundbar ist. Das Attribut Punkte des Objekts Geist beinhaltet die Punktezahl, die PacMan beim Zusammetreffen mit dem Geist erhalten kann, während lebendig angibt, ob der Geist noch am Leben ist.
Da Geist und PacMan ähnliche Attribute haben können sie von einem Objekt Person abgeleitet werden, das dann die gemeinsamen Attribute (x, y, Form und Farbe) erhält.

Nun gilt es die Methoden festzulegen.
Klasse Labyrinth:

Das Labyrinth muss man anlegen können, dann muss es gezeichnet werden. Das Labyrinth muss Auskunft geben, ob ein Feld frei ist, ob ein Punkt oder ein dicker Punkt auf dem Feld ist und ob noch Punkte vorhanden sind, sowie seine Größe angeben. Dann benötigt man noch eine Methode um ein einzelnes Feld zeichnen zu können sowie zum Löschen eines Punktes.

Folgendes Labyrinth soll entstehen (siehe nächste Seite)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
2	X	o	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	o	X
3	X	.	X	X	X	X	X	X	X	X	X	.	X	X	X	X	X	X	X	X	X	.	X
4	X	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	X	.	X
5	X	.	X	.	X	X	X	.	X	X	X	X	X	X	X	.	X	X	X	.	X	.	X
6	X	.	X	.	.	.	X	.	.	.	.	.	.	.	.	.	X	.	.	.	X	.	X
7	X	.	X	X	X	.	X	X	X	X	X	.	X	X	X	X	X	.	X	X	X	.	X
8	X	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	X
9	X	X	X	X	X	.	X	.	X	X	X	o	X	X	X	.	X	.	X	X	X	X	X
10	.	.	.	.	.	.	X	.	X						X	.	X	.	.	.	.	.	.
11	X	X	X	X	X	.	X	.	X		X	X	X		X	.	X	.	X	X	X	X	X
12	X	.	.	.	.	.	X	.	X						X	.	X	.	.	.	.	.	X
13	X	.	X	X	X	.	X	.	X	X	X	o	X	X	X	.	X	.	X	X	X	.	X
14	X	.	X	.	.	.	X	.	.	.	.	.	.	.	.	.	X	.	.	.	X	.	X
15	X	.	X	.	X	.	X	.	X	X	X	X	X	X	X	.	X	.	X	.	X	.	X
16	X	.	X	.	X	.	.	.	.	.	.		.	.	.	.	.	.	X	.	X	.	X
17	X	.	X	.	X	.	X	X	X	X	X	X	X	X	X	X	X	.	X	.	X	.	X
18	X	o	.	.	X	.	.	.	.	.	.	.	.	.	.	.	.	.	X	.	.	o	X
19	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

(X bedeutet Mauer, . ist eine normaler Punkt, o ein dicker Punkt)

Damit ergibt sich folgende Klasse:

Labyrinth
xGroesse yGroesse Plan Punkteanzahl
Init Zeige_Feld (x,y) Zeige_dich Sage_Weg_frei (x,y) Sage_Punkt (x,y) Sage_dicker_Punkt (x,y) Lösche_Punkt (x,y) Sage_Punkteanzahl

Klasse Spiel:
Das Spiel verwaltet den Punktestand. Dieser muss erhöht und ausgegeben werden können. Der Punktestand und die Anzahl der Leben wird ausgegeben. Bei der Init-Methode wird als Parameter die Anzahl der Leben eingegeben.

Damit erhält man folgende Klasse

Spiel
Punktestand Leben
Init (Zahl) Erhoehe_Punktestand (Zahl) Sage_Punktestand Reduziere_Leben Sage_Leben

Klasse Tastatur
Ein einfaches Objekt ist die Tastatur. Es hat nur die Attribute Taste und gedrückt. Diese beiden Attribute müssen initialisiert werden und ihre Werte müssen abfragbar sein.

Damit erhält man folgende Klasse

Tastatur
Taste gedrueckt
Init Sage_gedrueckt Sage_Taste

Klasse Figur
Die Objekte PacMan und Geist haben ähnliche Attribute und ähnliches Verhalten. Daher bietet sich an ein gemeinsames Vaterobjekt Figur zu entwerfen, von dem beide Objekte durch Vererbung abgeleitet werden.

Das Objekt Figur hat dann als Attribute x- und y- Koordinate, sowie eine Form. Außerdem eine bestimmte Farbe, die bei PacMan geändert werden kann. Die Objekte werden mit Init angelegt. Wie bei Asterix müssen sich die Figuren zeigen und verstecken können. Allerdings muss beim Verstecken, die ursprüngliche Markierung des Labyrinths wieder erscheinen. Auf diesen beiden Methoden baut dann Bewege_Dich, Lasse_Dich_Steuern, Folge und Fliehe_vor auf. Hier muss immer das Labyrinth gefragt werden, ob dieser Zug möglich ist. Weiter Methoden benötigt man zur Ausgabe der Attribute (Sage_x, Sage_y). Dazu müssen sie erfahren, ob er unverwundbar ist, genauso wie die Geister sagen können, ob sie noch am Leben sind. Des weiteren muss man feststellen können, ob sich zwei Figuren treffen, bzw. das Sterben der Objekte festgelegt werden. Die Geister müssen angeben können, wieviele Punkte man von ihnen erhält und ob sie noch am Leben sind. PacMan muss darüberhinaus verwundbar und unverwundbar werden können und seine Uhr (die die Zeit der Unverwundbarkeit misst) reduzieren können.

Alle Methoden werden bereits in dem Objekt Figur angelegt. Solche, die sich Attribute der abgeleiteten Objekte beziehen, nur als virtuelle Methoden.

Somit ergibt sich folgendes Objekt (bei dem manche Methoden virtuell sind).

Figur
x, y Form Farbe
Init (x, y, Form) Zeige_Dich Verstecke_Dich (Labyrinth) Sage_x Sage_y Bewege_Dich (dx, dy, Labyrinth) Lasse_dich_steuern (Taste, oben, unten, links, rechts, Labyrinth) Folge (Figur, Labyrinth) Fliehe_Vor (Figur, Labyrinth) Sage_unverwundbar Sage_am_Leben Sage_Punkte Trifft (Figur) Werde_unverwundbar Reduziere_Uhr Werde_verwundbar Stirbt (Labyrinth)

Die farbigen Methoden sind virtuell.

Klasse PacMan
PacMan hat als zusätzliches Attribut die Zeit, in der er unverwundbar ist. Hierzu benötigt man Methoden, mit denen man diesen Zustand ändern und angeben kann, sowie die Zeit verändern kann. Damit ergibt sich folgendes Objekt:

PacMan
unverwundbar Uhr
Init (x, y, Form) Sage_unverwundbar Reduziere_Uhr Werde_unverwundbar Werde_verwundbar Stirbt (Labyrinth)

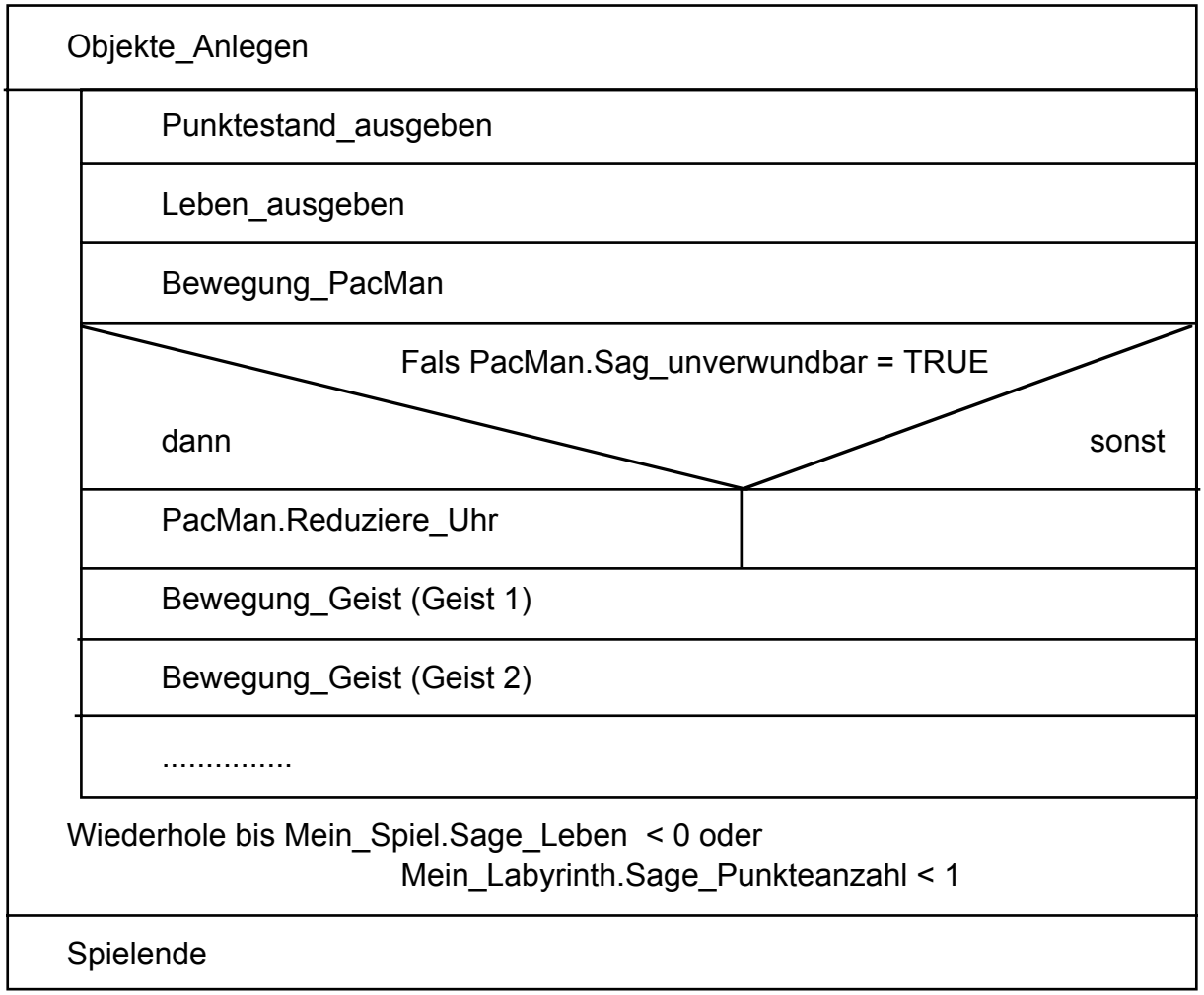
Klasse Geist
Die Geister haben die zusätzliche Attribute lebendig und die Anzahl der Punkte die man beim Zusammentreffen erhalten kann. Außerdem benötigt man noch Methoden, die die Anzahl der Punkte liefern und mit denen man Überprüfen kann, ob der Geist noch am Leben ist sowie eine Methode Stirbt.
Damit ergibt sich folgendes Objekt:

Geist
lebendig Punkte
Init (x, y, Form) Sage_Punkte Sage_am_Leben Stirbt (Labyrinth)

Nach dem Entwurf der Objekte kann nun mit Hilfe deren Attribute und Methoden das Programm verwirklicht werden.

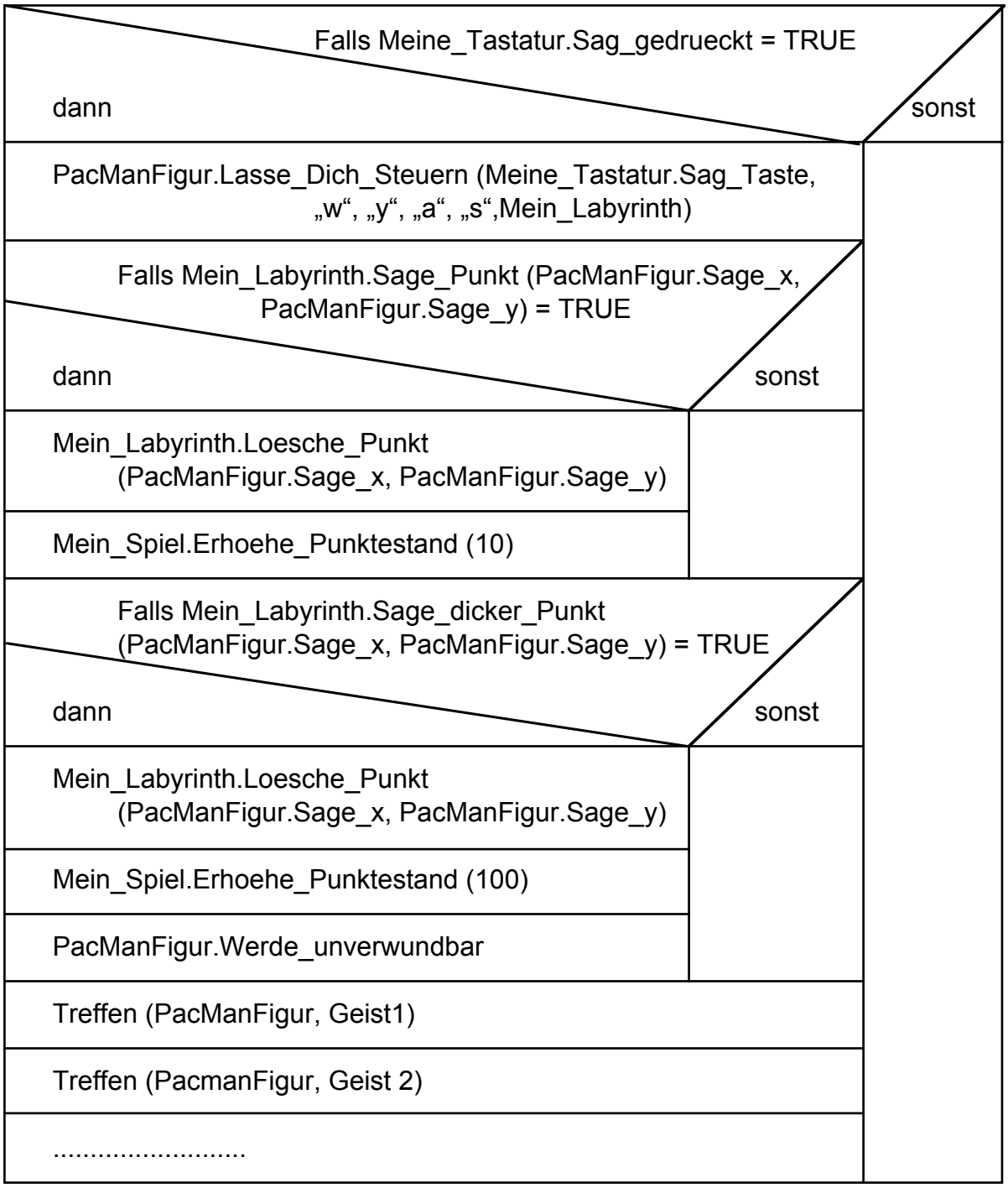
Das Hauptprogramm besteht zuerst aus dem Anlegen der Objekte und dann aus einer Wiederholungsanweisung, in der sich die einzelnen Objekt bewegen und deren Treffen behandelt werden muss.

Für das Hauptprogramm ergibt sich damit folgendes Struktogramm.



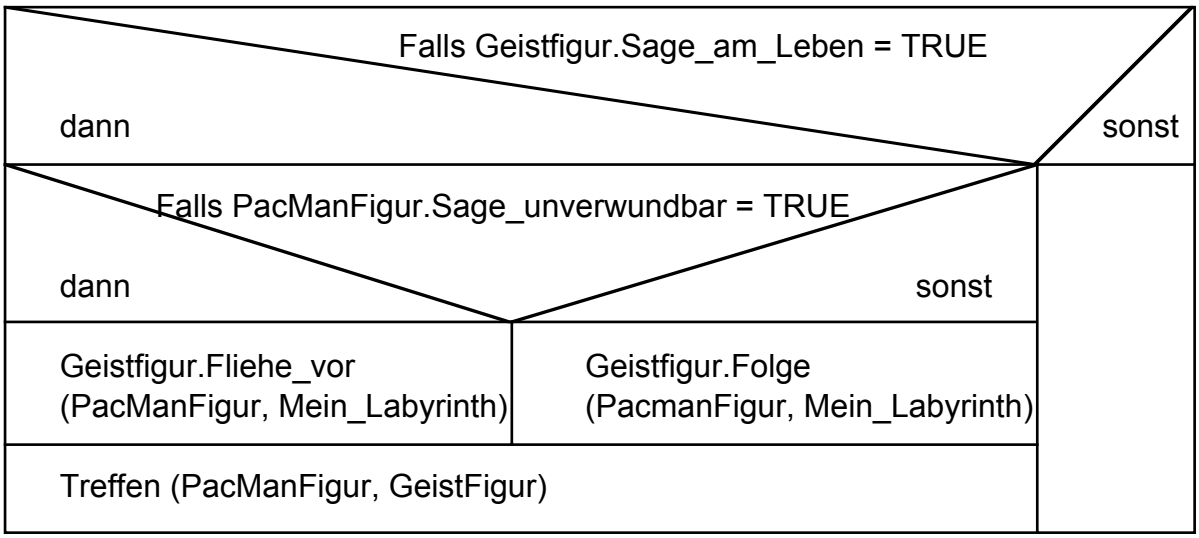
Die Bewegung PacMans wird in der Prozedur Bewegung_PacMan gesteuert. PacMan bewegt sich mit Hilfe der Tastatur. Hierbei kann er Punkte und dicke Punkte sammeln, sowie auf Geister treffen.

PROCEDURE Bewegung_PacMan



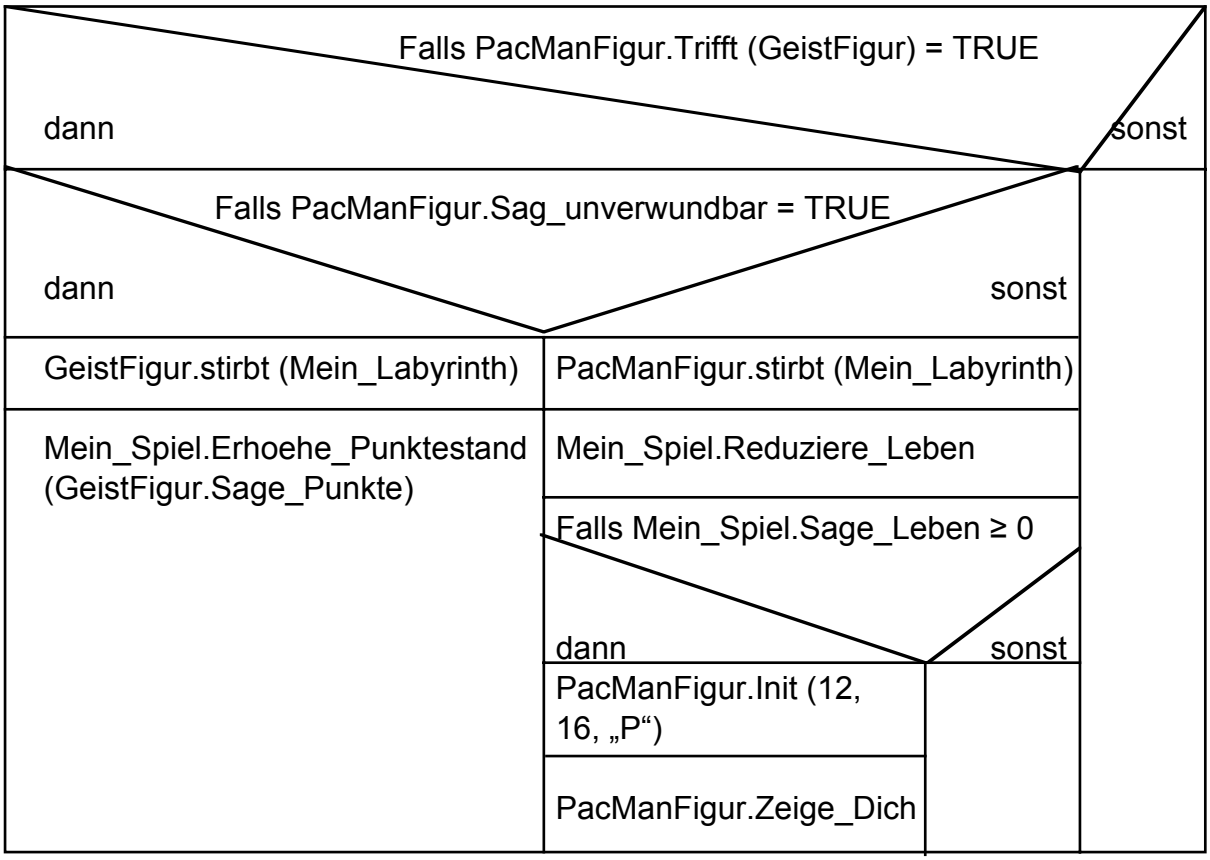
Die Bewegung der Geister erfolgt automatisch. Sie bewegen sich nur noch, wenn sie am Leben sind. Dann verfolgen sie oder fliehen vor PacMan, je nachdem ob dieser unverwundbar ist oder nicht. Auch hier muss das Treffen mit PacMan behandelt werden.

PROCEDURE Bewegung_Geist (Geistfigur : cGeist)



Nun fehlt nur noch die Behandlung des Aufeinandertreffens von PacMan und Geist. Ist PacMan unverwundbar, so stirbt der Geist. PacMan erhält zusätzliche Punkte. Im anderen Fall stirbt PacMan. Falls er noch ein Leben hat, wird er an der Startstelle neu geboren und das Spiel geht weiter.

PROCEDURE Treffen (PacManFigur : cPacMan, Geist Figur: cGeist)



Für eine Verbesserung des Projekts gibt es viele Möglichkeiten.

– Als erstes muss man die Geister bremsen, da man sonst, vor allem bei schnellen Rechnern keine Chance hat.

– Der Algorithmus für das Verfolgen ist nicht besonders effektiv.

– Die Geister könnten nach dem Tod nach einer gewissen Zeit wieder belebt werden.

– In PacMan-Spielen gibt es meist noch Früchte , die man fangen muss. Dadurch erhält man weitere Punkte und Leben.

– Auch weitere Level mit unterschiedlichen Labyrinthen sind möglich.

