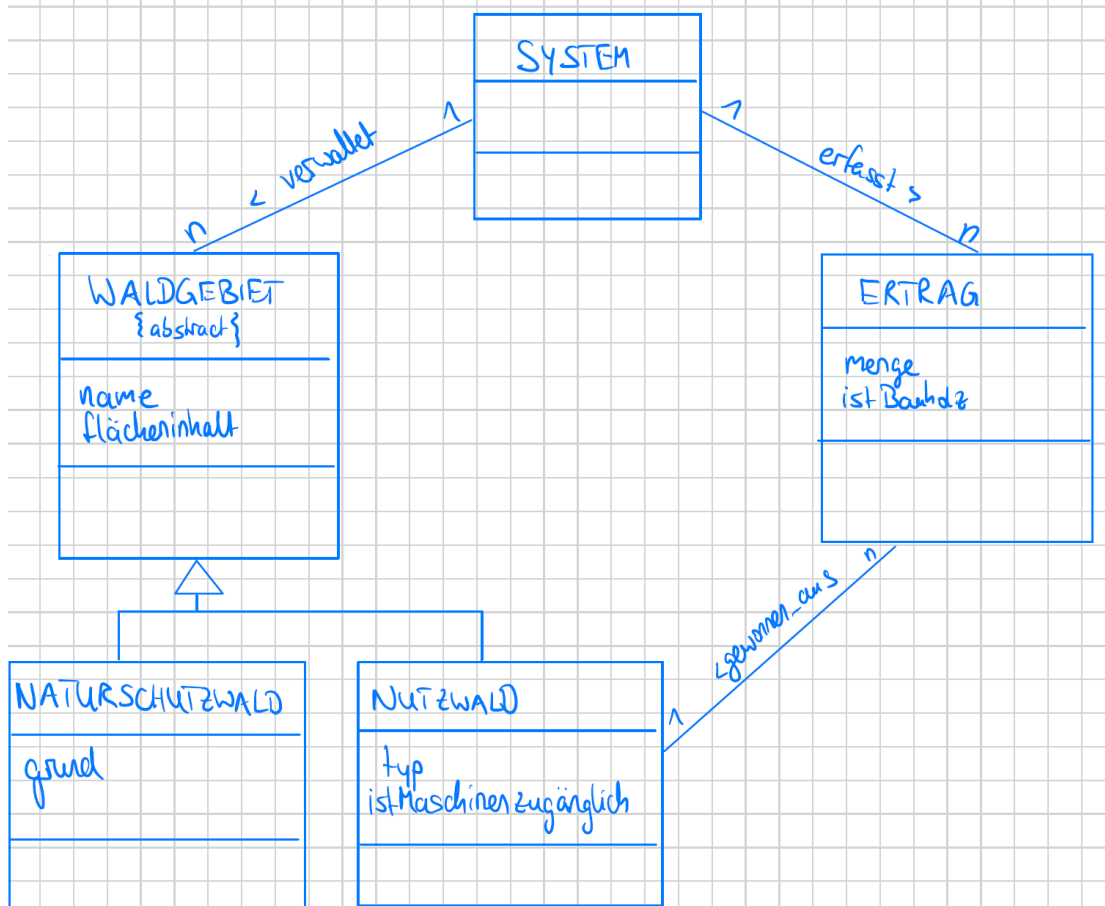


1a Klassendiagramm:

5

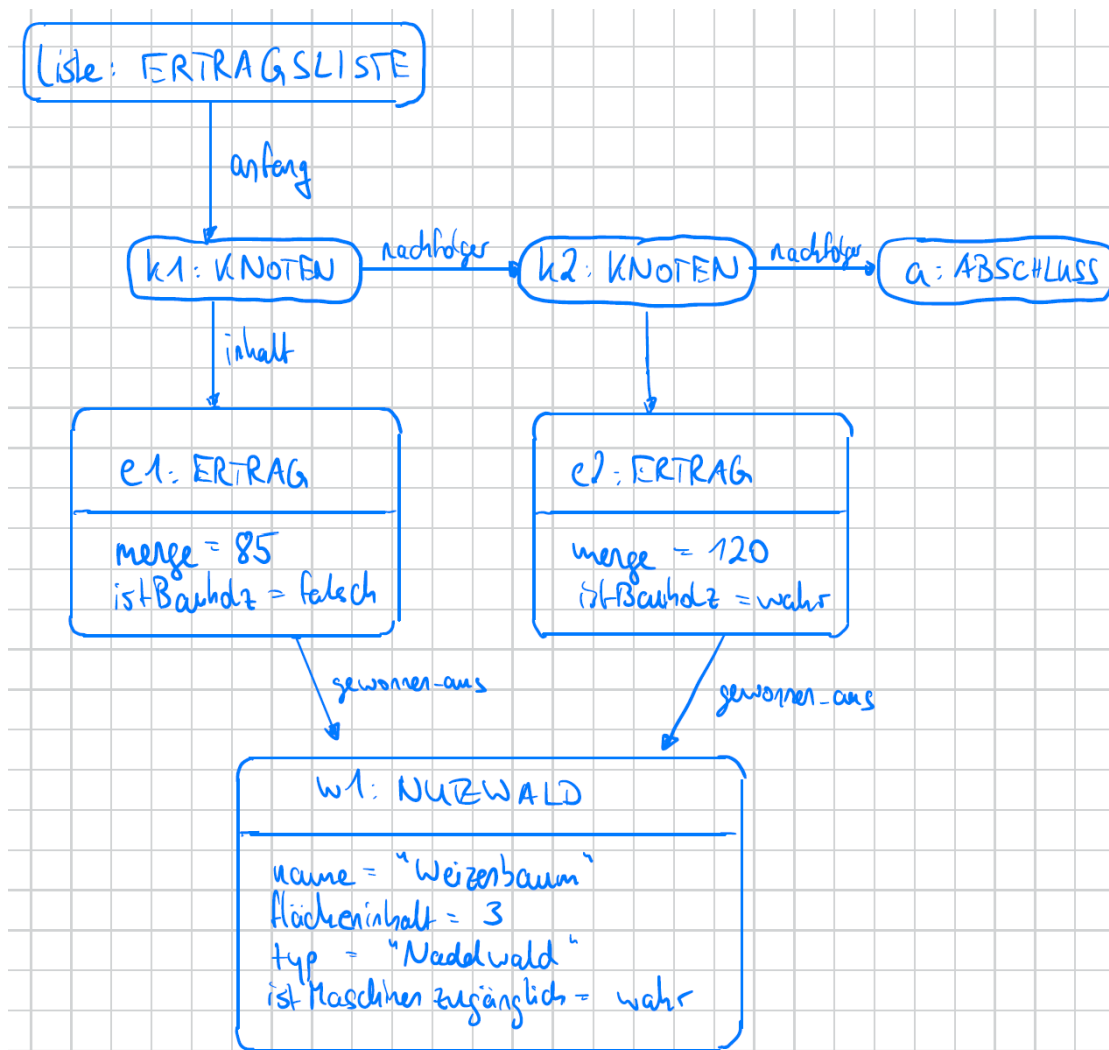


- 1b Vorteil Kompositum: Listenende mit Abschluss-Objekt. Dadurch werden weniger bedingte Anweisungen benötigt.
 Vorteil Entwurfsmuster: Bewährte Vorlagen für ähnliche und wiederkehrende Problemstellungen

2

1c Objektdiagramm:

3



1d Implementierung (Java):

9

ERTRAGSLISTE:

```

int bauholzGesamtmengeGeben(){
    return anfang.bholzGesGeben();
}

void etragEntfernen(Nutzwald nutzwald){
    anfang = anfang.ertragEntfernen(nutzwald);
}
  
```

LISTENELEMENT:

```

abstract int bholzGesGeben();

abstract LISTENELEMENT etragEntfernen(Nutzwald nutzwald);
  
```

KNOTEN:

```

int bholzGesGeben (){
    if(inhalt.istBauholzGeben()){
        return nachfolger.bholzGesGeben() + inhalt.mengeGeben();
    } else {
        return nachfolger.bholzGesGeben();
    }
}
  
```

```

LISTENELEMENT etragEntfernen(Nutzwald nutzwald){
    nachfolger = nachfolger.ertragEntfernen(nutzwald);
    if(inhalt.nutzwaldGeben() == nutzwald){
        return nachfolger;
    } else {
        return this;
    }
}

```

ABSCHLUSS:

```

int bholzGesGeben (){
    return 0;
}

```

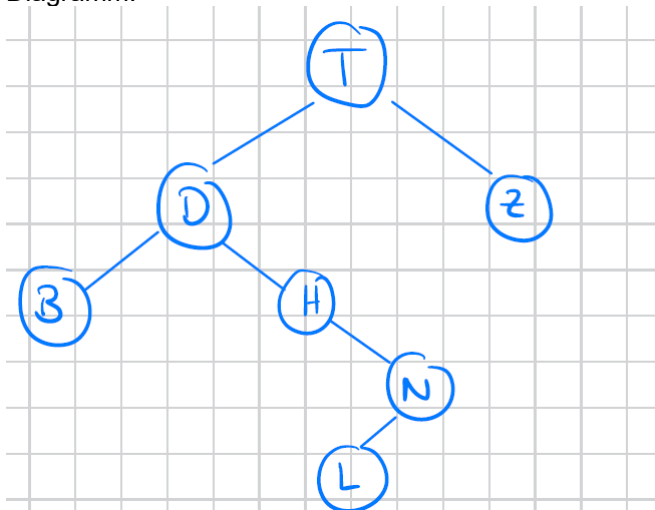
```

LISTENELEMENT etragEntfernen(Nutzwald nutzwald){
    return this;
}

```

2a Diagramm:

2



2b $L \rightarrow D \rightarrow B \rightarrow H \rightarrow T \rightarrow N \rightarrow Z$

1

2c $\text{Aufrunden}(\log_2 2056) = 12 \rightarrow 12 \text{ Ebenen}$

1

2d Implementierung (Java):

3

BINÄRBAUM:

```

int anzGeben(){
    return wurzel.anzGeben();
}

```

BAUMELEMENT:

```

abstract int anzGeben();

```

KNOTEN:

```
int anzGeben (){
    return linkerNachf.anzGeben() + rechterNachf.anzGeben()+1;
}
```

ABSCHLUSS:

```
int anzGeben (){
    return 0;
}
```

2e Implementierung (Java):

4

BINÄRBAUM:

```
boolean istEnthalten(String name){
    return wurzel.istEnt(name);
}
```

BAUMELEMENT:

```
abstract boolean istEnt(String name);
```

KNOTEN:

```
boolean istEnt(String name){
    int vgl = inhalt.nameGeben().vergleicheMit(name);
    if(vgl == 0){
        return true;
    } else if(vgl < 0) {
        return rechterNachf.istEnt(name);
    } else {
        return linkerNachf.istEnt(name);
    }
}
```

ABSCHLUSS:

```
boolean istEnt(String name){
    return false;
}
```

30

- 1 In der agilen Softwareentwicklung dienen Tasks als Arbeitseinheiten, in die ein User Story zerlegt wird. Durch iterative Zyklen kann man flexibel auf sich verändernde Anforderungen reagieren. 2

- 2a Beispiellösung für Python: 6

```
def tiefensucheStart(self, startBezeichner):
    start = self.knotennummerGeben(startBezeichner)
    for i in range(0, len(self.knoten)):
        self.knoten[i].besuchtSetzen(False)
    self.Tiefensuche(start)

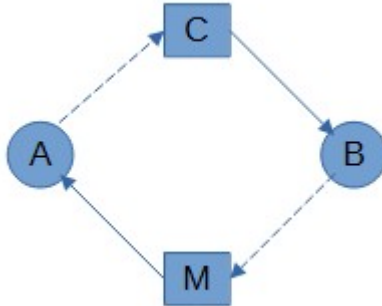
def tiefensuche(self, i):
    self.knoten[i].besuchtSetzen(True)
    print(self.knoten[i].bezeichnerGeben())
    for j in range(0, len(self.knoten)):
        if self.matrix[i][j]>0 and not self.knoten[j].istBesucht():
            self.tiefensuche(j)
```

Ein nach Ablauf des Algorithmus unbesuchter Knoten kann unabhängig von allen ausgegebenen Tasks bearbeitet werden.

- 2b In einem zyklischen Abhängigkeitsgraphen wartet jeder Task in einem Zyklus auf den Abschluss aller anderen Tasks im Zyklus. Damit kann ein Task erst begonnen werden, nachdem er selbst abgeschlossen ist. Das ist ein Widerspruch. 2

- 3a Alice hat exklusiven Zugriff „Model“, Bob auf das Betriebsmittel „Controler“. 2

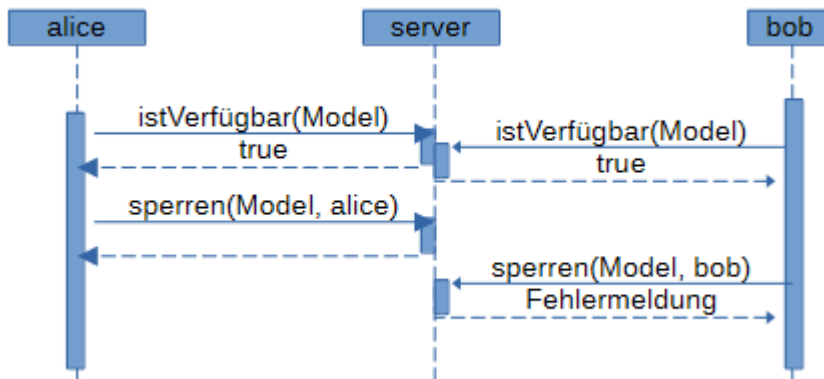
Beide fordern jeweils Zugriff anderen an.



auf das Betriebsmittel Betriebsmittel „Controler“. auf das Betriebsmittel des

- 3b Coffman-Bedingungen: wecl Ununterbrechbarkeit 4
- Ununterbrechbarkeit: Ein E freizugeben. Dies ließe sich führen, dass Entwickler sich
 - Halten und Warten: Ein En Zeit auf eine Ressource wartet. Dies kann der Entwickler selbständig erkennen und es ist eine praktikable Lösung.

- 3c Zunächst 4



überprüfen beide die Verfügbarkeit und erhalten positive Rückmeldung. Bei der Sperrung war Alice schneller und Bob erhält eine Fehlermeldung. Prüfung und Sperrung dürfen nur als Einheit ausgeführt werden und sollten durch einen Monitor geschützt werden.

- 4a Speicherzelle Inhalt 2
- | | |
|-------|-------|
| 5000 | 1234 |
| 1000 | 11234 |
| 11234 | 1 |
| 6000 | 1 |

- 4b In Zeile 5 und 6 wird überprüft, ob der Mitarbeiter die Nr. 1337 hat; in diesem Fall wird direkt zu „ok“ gesprungen, ohne dass dessen Zähler der Getränke inkrementiert wird. 2

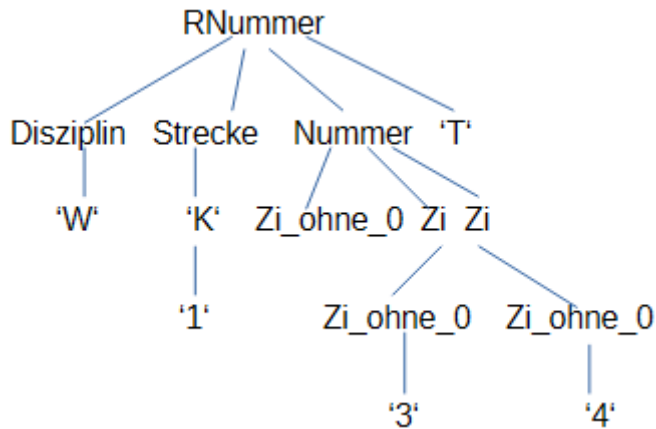
4c Fetch 1: Laden des Opcodes von load ins Befehlsregister, inkrementieren des Befehlszählers 2
Decode: Decodieren des Opcodes zum load-Befehl
Fetch 2: Laden des Operanden 5000 in das Befehlsregister, inkrementieren des Befehlszählers
Execute

4d start: load 5000 4
jeq start
addi 10000
store 1000
loadi 0
store 5000
load (1000)
jle start
subi 1
store (1000)
loadi 1
store 6000
jmp start

30

- 1a RNummer = Disziplin Strecke Nummer ['T'].
 Disziplin = 'W'|'L'.
 Strecke = 'K'|'V'.
 Nummer = Zi_ohne_0 [Zi][Zi][Zi].
 Zi_ohne_0 = '1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9'.
 Zi = Zi_ohne_0 |'0'.

4



- 1b WK10000
 Der Automat akzeptiert beliebig lange Startnummern, weil vierstellige nicht ausreichen.

1

- 1c `zustand = 1`
`def istRegNummer(eingabe):`
 `for zeichen in eingabe:`
 `zustandWechseln(zeichen)`
 `return (zustand == 4 or zustand == 5)`

`def zustandWechseln(zeichen):`
 `global zustand`
 `match(zustand):`
 `...`
 `case 4:`
 `if zeichen == 'T':`
 `zustand = 5`
 `elif zeichen in '0123456789':`
 `zustand = 4`
 `else: zustand = -1`
 `...`

6

- 1d L1 ist regulär, weil es eine endliche Anzahl an Wörtern in der Sprache gibt.
 L2 ist regulär, da es einen passenden erkennenden Automaten gibt.

2

- 2a Die Abbruchbedingung lautet (n gleich 1) oder (m gleich 1), d. h. das Gitter hat nur eine Spalte oder nur eine Zeile. Damit gibt es nur einen Weg, also muss 1 zurückgegeben werden.

2

- 2b $a(3,3) = a(2,3) + a(3,2) = a(1,3) + a(2,2) + a(2,2) + a(3,1) = 1 + a(1,2) + a(2,1) + a(1,2) + a(2,1) + 1 = 6$
 Anzahl rekursiver Aufrufe: 10

3

- 2c Zeile 3: Gib 0 zurück
 Zeile 5: Gib `anzahlAufrufeGeben(n-1, m) + anzahlAufrufeGeben(n, m-1) + 2` zurück

2

- 2d

3

	j=0	j=1	j=2	j=3
i=0	1	1	1	1
i=1	1	2	3	4
i=2	2	3	6	10

- 2e `anzahlWege[n-1][m-1]`

1

- 2f Quadratisches Laufzeitverhalten aufgrund der verschachtelten Wiederholung

2

- 2g Klaus bezieht sich auf das Halteproblem. Der Widerspruchsbeweis von Turing besagt, dass

4

es keine Maschine (Programm) geben kann, die für beliebige Methoden in Kombination mit einer Eingabe feststellen kann, ob die Methode unter der Eingabe terminiert. Klaus liegt falsch, da es eine solche Maschine für eine konkrete Methode durchaus geben kann.

30

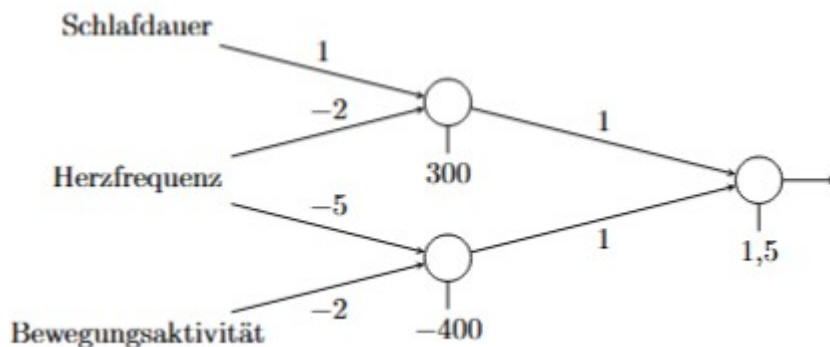
- 1a Neuronen links bilden die **Eingabeschicht**. Die Eingabeschicht empfängt externe Rohdaten und leitet diese unverändert weiter. Die mittleren drei Neuronen bilden die **verborgene Schicht**. Dort werden die Eingabedaten verarbeitet und Zusammenhänge erkannt. Das rechte Neuron bildet die **Ausgabeschicht**. In der Ausgabeschicht werden die Berechnungen der vorherigen Schicht in eine lesbare Antwort umgewandelt, welche die Eingabe klassifiziert.

- 1b Hinweis: Heaviside als Aktivierungsfunktion
- Mitte oben:** $6 \cdot 2 + 5 \cdot 8 = 52$ Da $52 \geq 40$ feuert (Ausgabe 1)
- Mitte mitte:** $6 \cdot -2 + 5 \cdot 10 = 38$ Da $38 < 60$ feuert nicht (Ausgabe 0)
- Mitte unten:** $6 \cdot 6 + 5 \cdot 2 = 46$ Da $46 \geq 35$ feuert (Ausgabe 1)

Ausgabe: $1 \cdot 4 + 0 \cdot (-1) + 1 \cdot 2 = 6$ Da $6 \geq 5,5$ feuert (Ausgabe 1)

Damit wurde bei dieser speziellen Eingabe der Tag als gesundheitsförderlich klassifiziert.

- 1c



- 2a Eine Kostenfunktion misst die mathematische Abweichung zwischen den Vorhersagen eines neuronalen Netzes und den tatsächlichen Zielwerten. Sie liefert eine einzige Kennzahl. Je kleiner diese ist, desto besser sind die Vorhersagen auf den Trainingsdaten. Ein typisches Beispiel ist MSE (Mean Squared Error). Der Wert der Kostenfunktion gibt an wie stark die Gewichte und Schwellenwerte beim Training angepasst werden müssen.
- 2b Es scheint als ob im dargelegten Beispiel die Trainingsdaten zu stark von der Realität abweichen. Sie sind eventuell veraltet, haben zu viele Ausreißer oder sind einfach falsch. Eine Anpassung der Hyperparameter kann dafür sorgen, dass bestimmte Muster der Trainingsdaten besser erkannt werden oder sich das Modell stärker oder weniger stark an den Trainingsdaten orientiert. Schlechte Trainingsdaten können damit aber nur bedingt ausgeglichen werden.
- 3 Integrität stellt sicher, dass Daten **korrekt, vollständig und unverändert** bleiben. Sie garantiert, dass Daten weder absichtlich (durch Angreifer) noch unabsichtlich (durch Systemfehler) unbemerkt modifiziert werden können. Im Beispiel sollten die Daten der Krankenkasse vor Manipulation geschützt gespeichert werden. Vertraulichkeit stellt sicher, dass Daten **nur von autorisierten Personen** eingesehen oder gelesen werden können. Unbefugte Personen, Systeme oder Prozesse dürfen keinen Zugriff auf diese Informationen haben. Da es sich um personenbezogene Daten handelt, sollte ausgeschlossen werden, dass Personen ohne berechtigtem Interesse Zugriff auf die Gesundheitsdaten haben.
- 4a K-Means ist ein Beispiel für **unsupervised learning**, da er noch unbekannte Muster in Rohdaten erkennen soll (ohne vorheriges Label).
- 4b Durch Pythagoras berechnet sich der Abstand zu den Zentren:

Aufgrund des geringeren Abstandes gehört P zu Cluster 2.

- 4c Da der neue Punkt N deutlich weiter von allen anderen Punkten entfernt ist als diese untereinander, werden zwei Cluster entstehen. Eines enthält alle anderen Punkte und eines nur Punkt N. 3

Bei der konkreten Berechnung wird N seinem Cluster zugeordnet. Alle anderen Punkte dem Cluster von Punkt P.

Im zweiten Schritt werden die Zentren aktualisiert. N bleibt Zentrum, für alle anderen Punkte wird das Zentrum neu berechnet. In weiteren Schritten ändert sich nichts mehr, da sich die Zuordnung durch das neue Zentrum nicht ändert.

- 4d Python: 2

```
def clusterzuordnungAktualisieren(self):

    geaendert = False

    for i in range(len(self.punkte)):
        bester_cluster = 0
        kleinsten_abstand = self.abstandGeben(self.punkte[i],
self.zentren[0])

        for j in range(len(self.zentren)):
            abstand = self.abstandGeben(self.punkte[i], self.zentren[j])

            if abstand < kleinsten_abstand:
                kleinsten_abstand = abstand
                bester_cluster = j

        if self.zuordnung[i] != bester_cluster:
            self.zuordnung[i] = bester_cluster
            geaendert = True

    return geaendert
```

- 4e Auf individueller Ebene gibt es für die Patienten einen finanziellen Anreiz der dafür spricht. 8
Dagegen spricht das Risiko, dass persönliche Gesundheitsdaten gespeichert und verarbeitet werden. Im schlimmsten Fall könnten die Daten missbraucht oder öffentlich werden.
Auf gesellschaftlicher Ebene fördern reale und aktuelle Gesundheitsdaten die medizinische Forschung und gönnen dadurch zu einer besseren Gesundheitsversorgung führen.
Allgemein sehen Kritiker eine Gefahr für einen Markt von Gesundheitsdaten, sollten Gesundheitsdaten konkrete Geldwerte zugeordnet werden.

40

