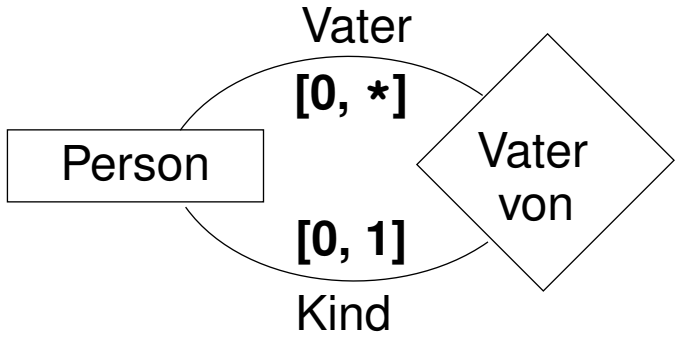
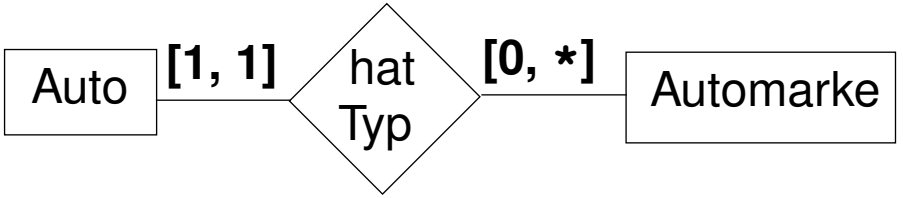
Hinweise zur Modellierung mit ER

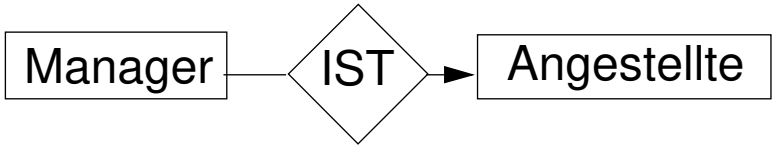
- In einem ER-Modell kommt **jede Entity-Menge nur einmal** vor.
- **Rollen** zu Relationen **angeben**, wo es nötig ist.
- Bedeutung der Kardinalitäten klarstellen.



- **Typ - Exemplar - Relationen** bewusst einsetzen.



- **Spezialisierung** sinnvoll einsetzen.



- Typ - Exemplar - Relation **nicht** mit Spezialisierung **verwechseln**

© 2001 bei Prof. Dr. Uwe Kastens

6. Modellierung von Abläufen
6.1 Endliche Automaten

Endlicher Automat:
Formaler Kalkül zur **Spezifikation von realen oder abstrakten Maschinen**. Sie

- reagieren auf **äußere Ereignisse**,
- ändern ihren **inneren Zustand**,
- produzieren ggf. **Ausgabe**.

Endliche Automaten werden **eingesetzt**, um

- das **Verhalten realer Maschinen** zu spezifizieren, z. B. Getränkeautomat,
- das **Verhalten von Software-Komponenten** zu spezifizieren,
  z. B. Reaktionen von Benutzungsoberflächen auf Bedienereignisse,
- **Sprachen zu spezifizieren**: Menge der Ereignis- oder Symbolfolgen, die der Automat akzeptiert, z. B. Schreibweise von Bezeichnern und Zahlwerten in Programmen

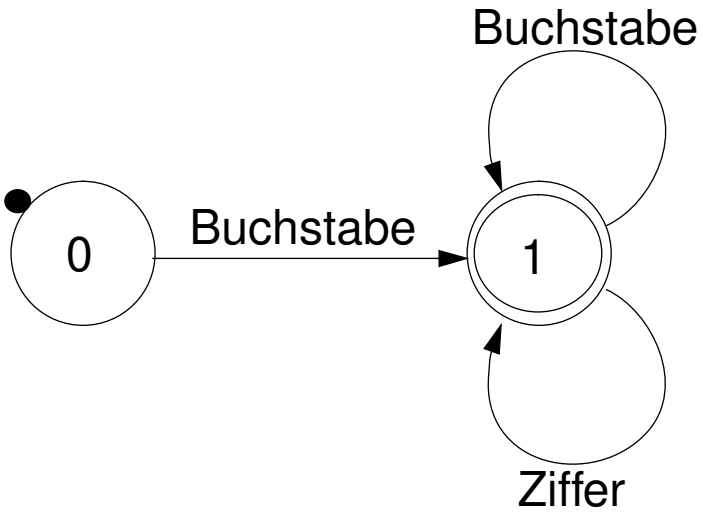
Zunächst definieren wir nur die **Eingabeverarbeitung** der Automaten;
das Erzeugen von **Ausgabe** fügen wir **später** hinzu.

© 2001 bei Prof. Dr. Uwe Kastens

Zwei einführende Beispiele

Endlicher Automat definiert eine **Sprache**, d. h. eine Menge von Wörtern.
Ein Wort ist eine Folge von Zeichen.

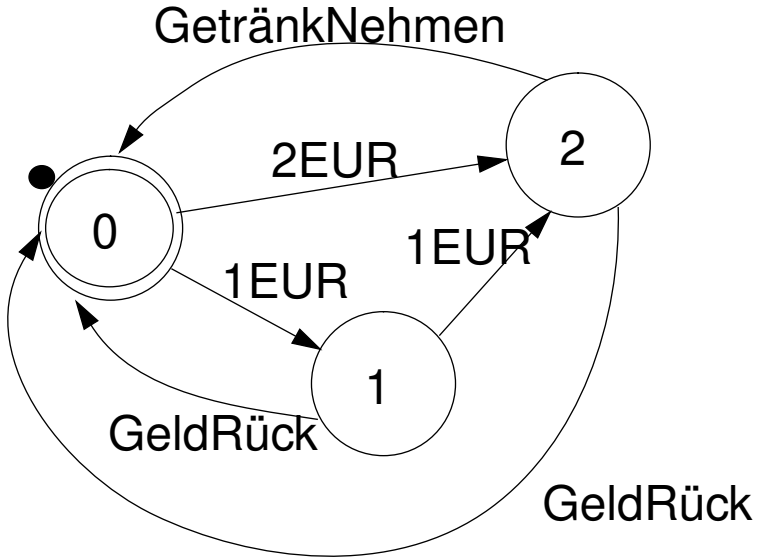
Hier: **Bezeichner** in Pascal-Programmen:



Akzeptiert Folgen von Buchstaben und Ziffern beginnend mit einem Buchstaben.

Endlicher Automat spezifiziert das **Verhalten einer Maschine**.

Hier: einfacher **Getränkeautomat**:



Akzeptiert Folgen von Ereignissen zur Bedienung eines Getränkeautomaten

Endliche Automaten können durch **gerichtete, markierte Graphen** dargestellt werden, **Ablaufgraphen**.

© 2002 bei Prof. Dr. Uwe Kastens

Alphabete

Alphabet: Eine **Menge von Zeichen** zur Bildung von Zeichenfolgen, häufig mit Σ bezeichnet.

Wir betrachten hier nur endliche Alphabete, z. B.

{0, 1}, {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, {a, b, ..., z}

Ein **Wort über einem Alphabet** Σ ist eine **Zeichenfolge** aus Σ^*

statt $(a_1, a_2, ..., a_n) \in \Sigma^*$ schreiben wir $a_1\ a_2\ ...\ a_n$, z. B. $10010 \in \{0, 1\}^*$

für die leere Folge schreiben wir ε (epsilon)

© 2002 bei Prof. Dr. Uwe Kastens

Reguläre Ausdrücke

R**eguläre Ausdrücke** beschreiben **Mengen von Worten**, die nach bestimmten Regeln aufgebaut sind. Seien F und G reguläre Ausdrücke, dann gilt

regulärer Ausdruck	**Menge von Worten**	**Erklärung**
a	$\{ a \}$	Zeichen a als Wort
ε	$\{ \varepsilon \}$	das leere Wort
F | G	$\{ f \mid f \in F \} \cup \{ g \mid g \in G \}$	Alternativen
F G	$\{ f\,g \mid f \in F, g \in G \}$	Zusammenfügen von Worten
F^n	$\{ f_1 f_2 ... f_n \mid \forall i \in \{1,..n\}: f_i \in F \}$	n Worte aus F
F^*	$\{ f_1 f_2 ... f_n \mid n \geq 0$ und $\forall i \in \{1,..n\}: f_i \in F \}$	Folgen von Worten aus F
F^+	$\{ f_1 f_2 ... f_n \mid n \geq 1$ und $\forall i \in \{1,..n\}: f_i \in F \}$	nicht-leere Folgen von Worten aus F
(F)	= F	Klammerung

Beispiele: $1^3 (1 \mid 0)^* 0^3$

Bezeichner = B (B | D)* mit B = a | b | ... | z und D = 0 | 1 | ... | 9

© 2002 bei Prof. Dr. Uwe Kastens

Deterministischer endlicher Automat

Deterministischer endlicher Automat (engl.: deterministic finite automaton, DFA):

Quintupel $A = (\Sigma, Q, \delta, q_0, F)$ mit

Σ	endliches **Eingabealphabet**
Q	endliche **Menge von Zuständen**
δ	**Übergangsfunktion** aus $Q \times \Sigma$ -> Q
$q_0 \in Q$	**Anfangszustand**
$F \subseteq Q$	**Menge der Endzustände** (akzeptierend)

Wir nennen $r = \delta(q, a)$ **Nachfolgezustand von q unter a**.

A heißt **deterministisch, weil** es **zu jedem Paar (q, a)**, mit $q \in Q$, $a \in \Sigma$,
höchstens einen Nachfolgezustand $\delta(q, a)$ gibt, d. h. δ ist eine **Funktion in Q.**

A heißt **vollständig, wenn** die **Übergangsfunktion** δ eine **totale** Funktion ist.

© 2002 bei Prof. Dr. Uwe Kastens

Gerichteter Graph zu endlichem Automaten

Knoten: **Zustände** des Automaten; Anfangszustand und Endzustände werden speziell markiert
Kanten: **Übergangsfunktion**, q -> r markiert mit a, genau dann wenn $\delta(q, a) = r$
Es gibt Kanten, die sich nur durch ihre Markierung unterscheiden, deshalb: **Multigraph**

Beispiele von Mod-6.2:

Σ := Menge der ASCII-Zeichen

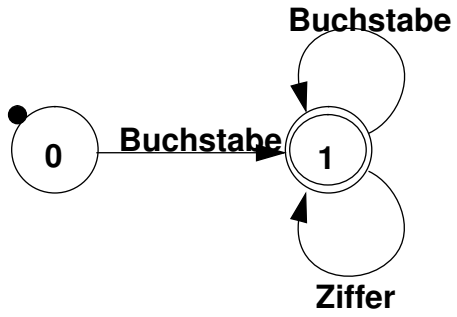
Q := {0, 1}

δ :=

	a...zA...Z	0...9	sonstige
0	1		
1	1	1	

$q_0 = 0$

F = {1}



Buchstabe, Ziffer
sind Namen reg. Ausdrücke

Σ := {**1EUR, 2EUR, GeldRück, GetränkNehmen**}

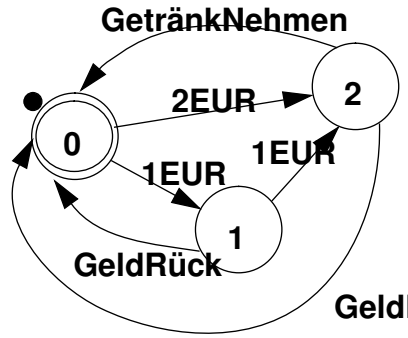
Q := {0, 1, 2}

δ :=

	1EUR	**2EUR**	**GeldRück**	**GetränkNehmen**
0	**1**	**2**		
1	**2**		**0**	
2			**0**	**0**

$q_0 = 0$

F = {0}



© 2002 bei Prof. Dr. Uwe Kastens

Akzeptierte Sprache

Die Zeichen einer Zeichenfolge bewirken nacheinander Zustandsübergänge in Automaten.
Zustandsübergangsfunktion erweitert für Zeichenfolgen:

Sei $\delta : Q \times \Sigma \rightarrow Q$ eine **Übergangsfunktion für Zeichen**,
dann ist $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ eine **Übergangsfunktion für Wörter**, rekursiv definiert:

- Übergang mit dem **leeren Wort**: $\hat{\delta}(q, \varepsilon) = q$ für alle $q \in Q$
- Übergang mit dem **Wort wa**: $\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$ für alle $q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$

Statt $\hat{\delta}$ schreiben wir meist auch δ.

Sei $A = (\Sigma, Q, \delta, q_0, F)$ ein deterministischer endlicher Automat und $w \in \Sigma^*$.
A akzeptiert das Wort w genau dann, wenn $\delta(q_0, w) \in F$.
Die Menge $L(A) := \{ w \in \Sigma^* \mid \delta(q_0, w) \in F \}$ heißt die **von A akzeptierte Sprache**.

Beispiele für Sprachen, die von endlichen Automaten akzeptiert werden können:

$L_1 = \bigcup_{n, m \in \mathbb{N}} a^n b^m$ $\qquad$ $L_2 = \Sigma^*$

Es gibt keinen endlichen Automaten, der $L_3 = \bigcup_{n \in \mathbb{N}} a^n b^n$ akzeptiert.

© 2002 bei Prof. Dr. Uwe Kastens

Nicht-deterministischer Automat

Nicht-deterministisch (allgemein) :

Es gibt mehrere Möglichkeiten der Entscheidung bzw. der Fortsetzung,
es ist aber nicht festgelegt, welche gewählt wird.

Nicht-deterministischer endlicher Automat:

Die **Übergangsfunktion** δ kann einen Zustand q und ein Eingabezeichen a auf **mehrere Nachfolgezustände** abbilden $\delta : Q \times \Sigma \rightarrow \text{Pow}(Q)$.
Welcher gewählt wird, ist nicht festgelegt.
Σ , Q, q_0, F sind wie für deterministische endliche Automaten definiert.

Erweiterung von δ auf Zeichenfolgen:

Sei $A = (\Sigma, Q, \delta, q_0, F)$ ein nicht-deterministischer endlicher Automat; dann ist $\hat{\delta}$ definiert:

- Übergang mit dem **leeren Wort**: $\hat{\delta}(q, \varepsilon) = \{ q \}$ für alle $q \in Q$
- Übergang mit dem **Wort wa**: $\hat{\delta}(q, wa) = \{q' \in Q \mid \exists p \in \hat{\delta}(q, w): q' \in \delta(p, a)\}$
  für alle $q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$,
  d. h. **die Menge aller Zustände, die man von q mit wa erreichen kann**

Wir schreiben meist δ für $\hat{\delta}$

Ein nicht-deterministischer endlicher Automat A **akzeptiert** ein Wort w gdw. $\delta(q_0, w) \cap F \neq \emptyset$

$L(A) = \{w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset \}$ ist **die von A akzeptierte Sprache**.

© 2002 bei Prof. Dr. Uwe Kastens

Nicht-deterministische und deterministische Automaten

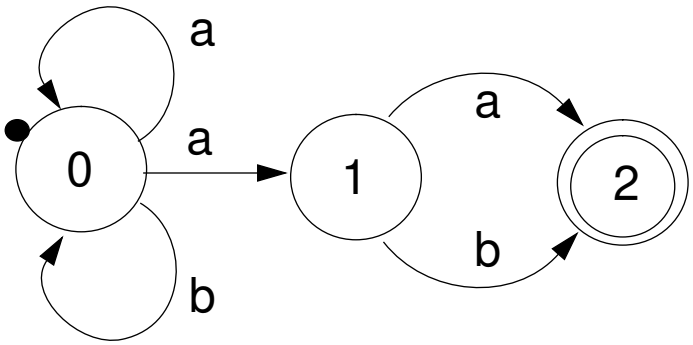
Satz: Sei L(A) die Sprache eines nicht-deterministischen Automaten.
Dann gibt es einen deterministischen Automaten, der L(A) akzeptiert.

Man kann **aus einem nicht-deterministischen Automaten A** = (Σ , Q , δ, q_0 , F)
einen **deterministischen A'** = (Σ, Q', δ', q_0', F') systematisch **konstruieren**:

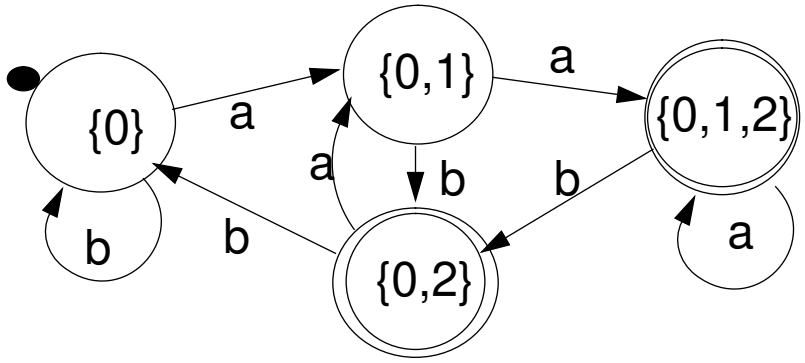
Jeder **Zustand aus Q' repräsentiert eine Menge von Zuständen aus Q, d. h. Q' $\subseteq$ Pow(Q)**

Beispiel:

nicht-deterministisch A



deterministisch A'



Die Zahl der Zustände kann sich dabei **exponentiell** vergrößern.

© 2002 bei Prof. Dr. Uwe Kastens

Konstruktion deterministischer Automaten

Sei A ein **nicht-deterministischer Automate A** = (Σ , Q , δ, q_0 , F) daraus wird ein **deterministischer Automat A'** = (Σ, Q', δ', q_0', F') systematisch **konstruiert**:

Jeder **Zustand aus Q' repräsentiert eine Menge von Zuständen aus Q, d. h. Q' $\subseteq$ Pow(Q)**

Konstruktionsschritte:

1. **Anfangszustand**: $q_0' = \{q_0\}$
2. Wähle einen schon konstruierten Zustand $q' \in Q'$
   wähle ein Zeichen $a \in \Sigma$

   berechne $r' = \delta'(q', a) = \bigcup_{q \in q'} \delta(q, a)$

   d. h. r' repräsentiert die Vereinigung aller Zustände, die in A von q unter a erreicht werden.
   r' wird **Zustand in Q'** und $\delta'(q', a) = r'$ wird **Übergang in** δ'.
3. **Wiederhole (2) bis keine neuen Zustände oder Übergänge** mehr konstruiert werden können.
4. **Endzustände**: $F' = \{q' \in Q' \mid q' \cap F \neq \emptyset\}$
   d. h. q' ist Endzustand, wenn seine Zustandsmenge einen Endzustand von A enthält.

© 2002 bei Prof. Dr. Uwe Kastens

Endliche Automaten mit Ausgabe

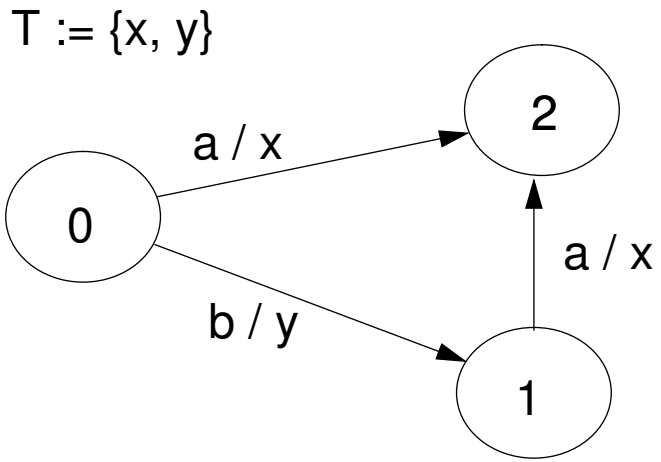
Man kann mit endlichen Automaten auch **Reaktionen der modellierten Maschine** spezifizieren: **Automaten mit Ausgabe**.

Wir erweitern den Automaten um ein **endliches Ausgabealphabet T** und um eine Ausgabefunktion. Es gibt 2 Varianten für die Ausgabefunktion:

Mealy-Automat:

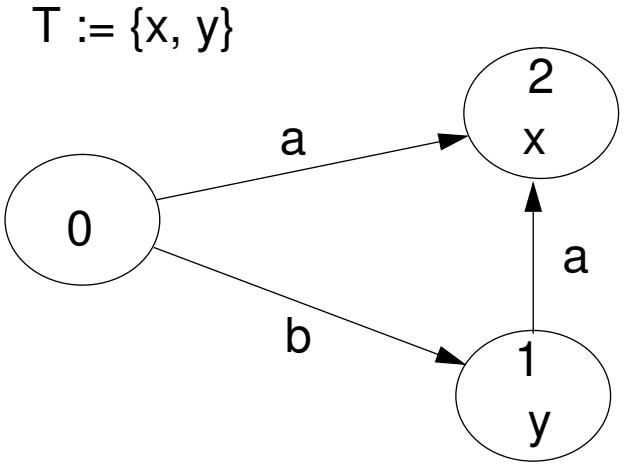
Eine Ausgabefunktion $\lambda : Q \times \Sigma \rightarrow T^*$ ordnet den **Zustandsübergängen** jeweils ein **Wort über dem Ausgabealphabet** zu.

Graphische Notation:



Moore-Automat:

Eine Ausgabefunktion $\mu : Q \rightarrow T^*$ ordnet den **Zuständen** jeweils ein **Wort über dem Ausgabealphabet** zu. Es wird bei Erreichen des Zustands ausgegeben.



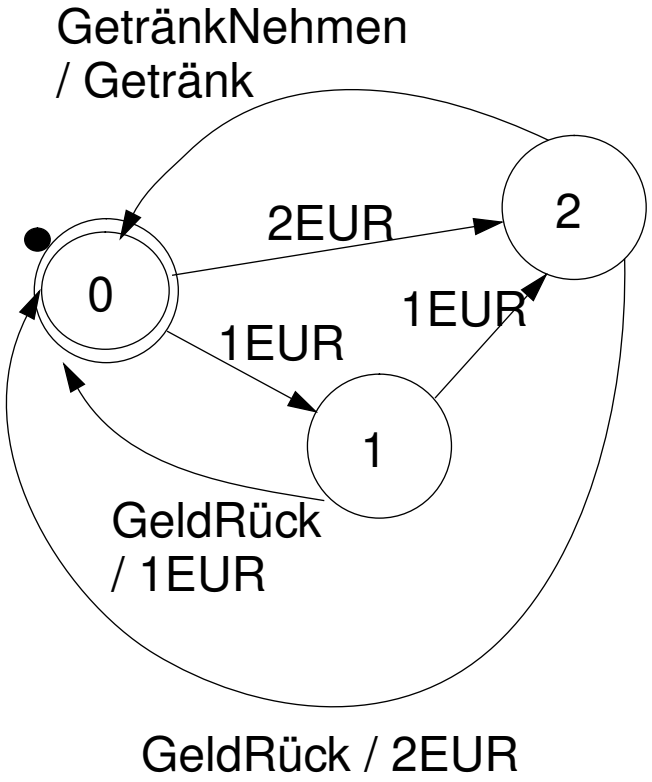
Ein **Mealy-Automat kann die Ausgabe feiner differenzieren** als ein Moore-Automat.

© 2002 bei Prof. Dr. Uwe Kastens

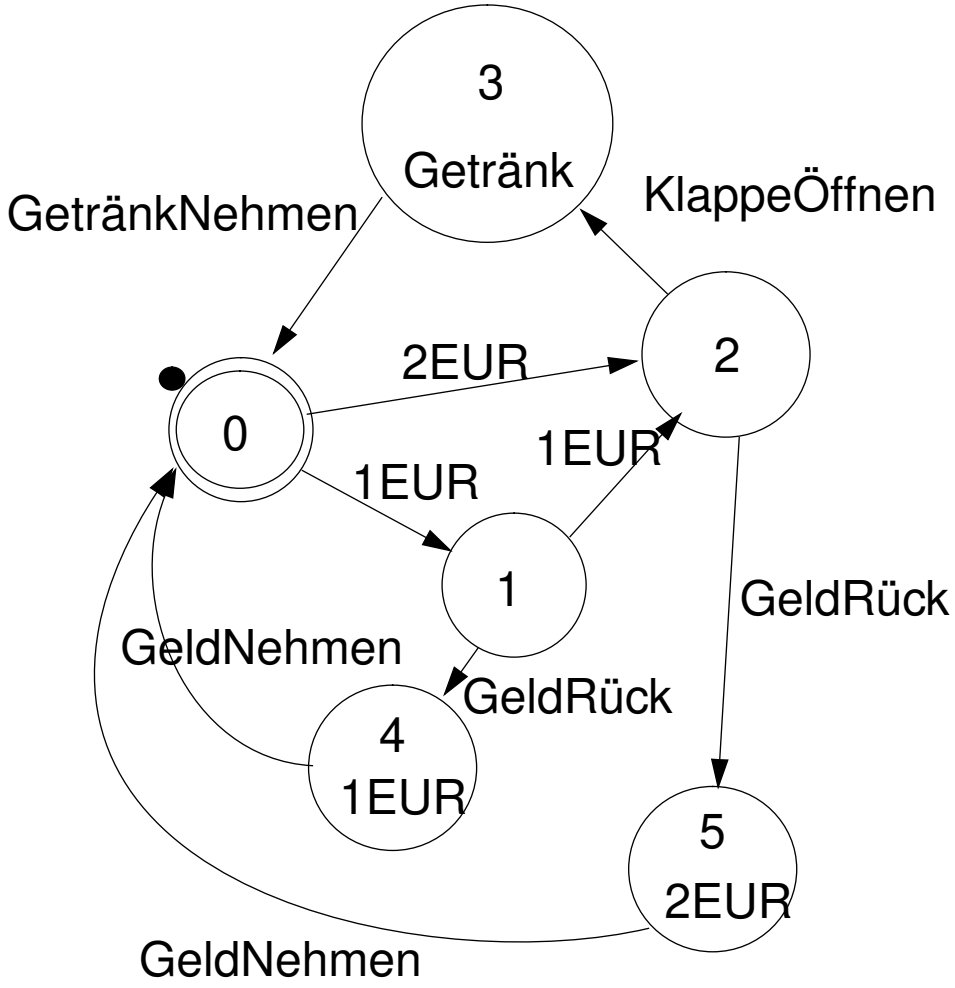
Beispiele für endliche Automaten mit Ausgabe

Die Spezifikation des Getränkeautomaten aus Mod-6.2 wird mit Ausgabe versehen:

Mealy-Automat

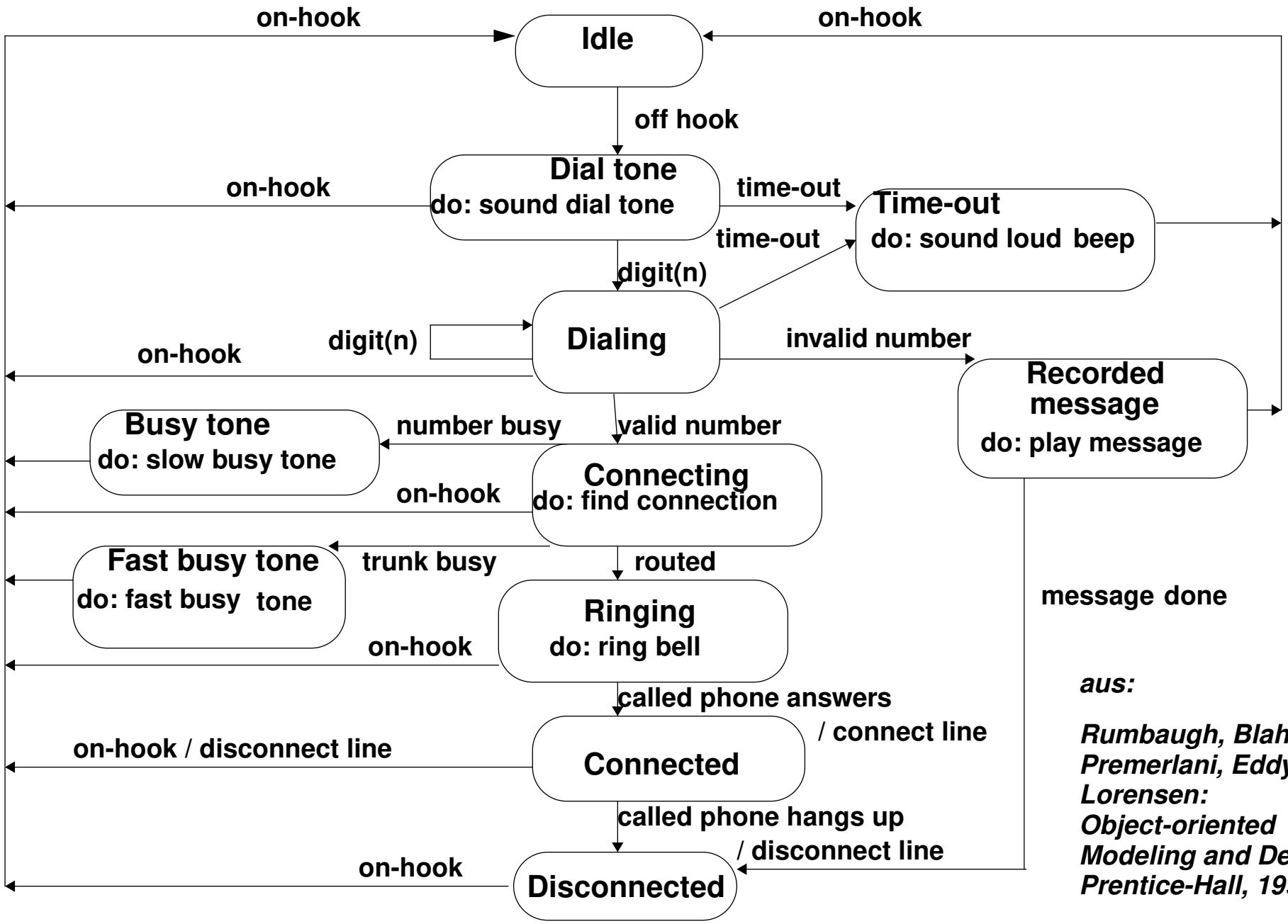


Moore-Automat



© 2002 bei Prof. Dr. Uwe Kastens

Endlicher Automat zur Telefonbedienung



aus:

Rumbaugh, Blaha, Premerlani, Eddy, Lorensen: Object-oriented Modeling and Design, Prentice-Hall, 1991

© 2001 bei Prof. Dr. Uwe Kastens