Diplomarbeit

Produktbedienungsanleitungen zur Charakterisierung austauschbarer Services

Jan Bretschneider

13. März 2007

Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät II
Institut für Informatik

Gutachter:
Prof. Dr. Karsten Wolf
Prof. Dr. Wolfgang Reisig

Zusammenfassung

Unternehmen sind bestrebt, immer mehr Geschäfte mit ihren Kunden teilweise oder vollständig automatisiert abzuwickeln. In diesem Bestreben machen sie mehr und mehr Gebrauch von der serviceorientierten Architektur (SOA). Grundbaustein der SOA ist der Service, der eine von einem Unternehmen angebotene Dienstleistung oder Funktionalität über eine wohldefinierte Schnittstelle bereitstellt und von Kunden oder Services anderer Unternehmen verwendet werden kann. Damit wir zwei Services als sinnvoll miteinander interagierend bezeichnen können, müssen sie verschiedene Mindestanforderungen erfüllen. Auf Grundlage dieser Mindestanforderungen können wir für jeden gegebenen Service eine Bedienungsanleitung konstruieren, die alle sinnvoll mit ihm interagierenden Services charakterisiert.

Auch tritt die Frage auf, gegen welche Services ein Service ausgetauscht werden kann, so dass alle Services, die mit dem alten sinnvoll interagieren konnten, auch mit dem neuen sinnvoll interagieren können.

Diesen allgemeinen Austauschbarkeitsbegriff parametrisieren wir in der vorliegenden Arbeit und beschäftigen uns mit dem Fall, dass durch den Austausch eines Services nur eine bestimmte Menge von Services unberührt bleiben soll, weil dies eine größere Freiheit in der Wahl des austauschenden Service erlaubt. Wir werden die Menge der Services, gegen die sich ein bestimmter Service bezüglich einer gegebenen Menge von Services austauschen lässt, mit Hilfe des Konzepts der Bedienungsanleitungen genau charakterisieren.

Inhaltsverzeichnis

1 Einleitung

Wie in den meisten Bereichen schreitet die Automatisierung auch in der Geschäftswelt weiter fort. Unternehmen sind daran interessiert, mehr und mehr Geschäfte mit ihren Kunden teilweise oder vollständig automatisiert abzuwickeln, um Zeit, Arbeitskraft und Kosten zu sparen.

In diesem Bestreben unterstützt sie die serviceorientierte Architektur (SOA) [Got00], von der Unternehmen deswegen zunehmend Gebrauch machen. Grundbaustein der SOA ist der *Service*, der eine Dienstleistung eines Unternehmens hinter einer wohldefinierten Schnittstelle kapselt. Innerhalb der SOA können Unternehmen ihre Services anbieten, und Kunden angebotene Services auffinden und verwenden. Ein Service ist dabei üblicherweise an einem eindeutig spezifizierten Ort auffindbar und wird benutzt, indem man Nachrichten mit ihm austauscht.

Da ein Service zur Erfüllung seiner Aufgabe nicht nur andere Services verwenden *kann*, sondern im Sinne der Automatisierung auch *soll*, ergeben sich Netzwerke von kommunizierenden Services. Folglich ist beim Entwurf von Services genau darauf zu achten, dass sie so miteinander interagieren, dass jeder Service stets zuverlässig die von ihm verlangten Aufgaben erledigen kann.

Damit zwei Services miteinander interagieren können, müssen als erstes ihre Schnittstellen zusammenpassen. Das bedeutet, dass sich zwei Services gegenseitig nur Nachrichten senden können dürfen, die der jeweils andere auch empfangen kann. In dieser Arbeit abstrahieren wir von Daten in den Nachrichten und unterscheiden nur Nachrichtentypen. Services, deren Schnittstellen zusammenpassen, nennen wir *Partner*.

Damit zwei Partner auch *sinnvoll* miteinander interagieren können, genügt es nicht, dass ihre Schnittstellen zusammenpassen. Daran, was sinnvoll interagieren bedeutet, lassen sich verschiedenste Anforderungen stellen. In dieser Arbeit stellen wir wie in [MRS05] zwei Mindestanforderungen an sinnvolle Interaktion. Erstens, dass die beiden Services zusammen in keinen Deadlock geraten können dürfen, und zweitens, dass am Ende ihrer Interaktion alle gesendeten Nachrichten auch empfangen worden sein müssen. Einen Service A, der mit einem anderen Service B gemäß dieser Mindestanforderungen kommuniziert, nennen wir *Strategie von B* [MRS05].

Mögliche Kunden, die einen Service R entworfen haben, möchten über einen angebotenen Service P gern wissen, ob R sinnvoll mit P interagieren kann, das heißt, ob R eine Strategie von P ist. Zu diesem Zweck kann der Anbieter des Service P eine *Bedienungsanleitung* [MRS05] von P veröffentlichen. Diese Bedienungsanleitung charakterisiert auf kompakte Weise die Menge aller Strategien von P. Anhand der Bedienungsanleitung von P kann ein möglicher Kunde also ablesen, wie sein Service R sinnvoll mit P kommunizieren kann. Alternativ kann er aus der Bedienungsanleitung von P auch gezielt eine Strategie von P konstruieren. Die Bedienungsanleitung von P wird zwar mit dem Wissen über die interne Struktur von P konstruiert, sie gibt aber nur so viel dieses Wissens preis, wie nötig ist, um sinnvoll mit P interagieren zu können. Dies kommt dem Anbieter von P zu gute, wenn er die interne Struktur von P weitgehend geheim halten möchte.

Unternehmen werden ihre angebotenen Services manchmal abändern oder gegen andere Services austauschen wollen oder müssen. Der Austausch eines Service sollte möglichst unbemerkt von seinen Benutzern geschehen können. Hinsichtlich der oben formulierten Mindestanforderungen an sinnvolle Interaktion zwischen Services stellt sich also die Frage, durch welchen neuen Service P' man einen alten Service P *austauschen* kann, so dass alle möglichen Strategien von P auch Strategien von P' bleiben [Mar03]. In dieser Arbeit beschränken wir uns dabei auf den Austausch während der Entwurfszeit von Services und nicht während ihrer Laufzeit.

Der oben genannte Austauschbarkeitsbegriff ist zwar effektiv in der Bewahrung von Strategien. Er ist aber auch restriktiv in dem Sinne, dass er nur wenig Änderungen am Service P zulässt, da *alle* möglichen Strategien von P erhalten bleiben müssen.

Es ist jedoch vorstellbar, dass Unternehmen zwar oft einen Service gegen einen anderen austauschen wollen, gleichzeitig aber nicht gewillt sind, alle theoretisch denkbaren Strategien dieses Service zu erhalten. Stattdessen könnte es ihnen genügen, wenn nur diejenigen Strategien erhalten bleiben, die zurzeit tatsächlich ihren Service benutzen. Also ergibt sich die Frage, durch welchen neuen Service P' man einen alten Service P bezüglich einer Menge $\mathcal{R}$ von Strategien von P austauschen kann, so dass jede Strategie von P aus $\mathcal{R}$ auch Strategie von P' ist. Genau diese Frage ist Gegenstand der vorliegenden Arbeit. Die Menge $\mathcal{R}$ ist dabei ein Parameter für den oben genannten allgemeinen Austauschbarkeitsbegriff.

Um diese Frage zu lösen, führen wir in Kapitel 2 zunächst die benötigten bereits existierenden Grundbegriffe und -zusammenhänge ein. Wir beginnen beim Begriff des Service, beschäftigen uns danach mit der Interaktion von Services und erläutern schließlich die Konstruktion ihrer Bedienungsanleitungen.

Anschließend schlagen wir in Kapitel 3 ein Produkt auf Bedienungsanleitungen vor, mit dem wir den Durchschnitt der Strategien einer gegebenen Menge von Services charakterisieren können.

Dieses Produkt wird uns in Kapitel 4 helfen, die oben genannte Austauschbarkeitsfrage bezüglich einer Menge von Strategien zu beantworten, nachdem wir den dazugehörigen parametrisierten Austauschbarkeitsbegriff formalisiert haben.

In Kapitel 5 werden wir uns mit einer weiteren Anwendungsmöglichkeit von Produktbedienungsanleitungen beschäftigen. Dort werden wir die Bedienungsanleitung eines gegebenen Serviceautomaten konstruieren, indem wir ihn zuerst in Teilautomaten zerlegen und dann das Produkt der Bedienungsanleitungen dieser Teilautomaten bilden.

Zu guter Letzt geben wir in Kapitel 6 eine Zusammenfassung der vorliegenden Arbeit und einen Ausblick auf offen gebliebene und weiterführende Fragen.

2 Grundlagen

In der Einleitung haben wir bereits erwähnt, dass der Begriff des Service der Grundbaustein für die serviceorientierte Architektur (SOA) [Got00] ist. Um die in dieser Arbeit gestellte Frage zur Austauschbarkeit von Services untersuchen zu können, müssen wir zuerst einen Formalismus finden, in dem wir Services samt ihres Verhaltens und die Kommunikation zwischen Services nicht nur geeignet modellieren, sondern auch analysieren können. In diesem Formalismus müssen wir auch ausdrücken können, was *sinnvolle* Kommunikation und Austauschbarkeit von Services bedeutet.

In der Literatur gibt es mehrere Ansätze, Services zu modellieren. Zwar lassen sich kommunizierende Services mit Hilfe von Prozessalgebren beschreiben und analysieren, und es gibt Arbeiten, die zu diesem Zweck CCS [BCPV04], LOTOS [Fer04] oder den π-Kalkül [RKM04] nutzen.

Zur Modellierung und Analyse von Geschäftsprozessen haben sich allerdings die auf Petrinetzen [Rei86] basierenden Workflow-Netze [Aal98] von van der Aalst durchgesetzt. Mit der in [Aal98] definierten Soundness-Eigenschaft lässt sich zudem überprüfen, wann ein Geschäftsprozess sinnvoll modelliert ist. Da Workflow-Netze keine Kommunikation modellieren, schlägt van der Aalst in [Aal99] organisationsübergreifende Workflow-Netze auf der Basis von Message Sequence Charts und Petrinetzen vor und bietet angepasste Soundness-Kriterien an.

Alternativ dazu schlagen Massuthe et. al. in [MRS05] die auf den Workflow-Netzen aus [Aal98] basierenden offenen Workflownetze (oWFNs), eine spezielle Klasse von Petrinetzen, zur Modellierung von kommunizierenden Services vor. Darin nennen sie ein oWFN P *Strategie* eines oWFNs R, wenn beide sinnvoll miteinander interagieren, und entwickeln für azyklische oWFNs den Begriff der *Bedienungsanleitung*, die alle Strategien eines Service auf kompakte Weise charakterisiert. In [LMW06] verwenden sie Serviceautomaten zur Modellierung von Services und erweitern ihre Arbeit auf zyklische, nichtdeterministische Services.

Aufbauend auf dem in [LMW06] errichteten Begriffsgerüst werden wir in Kapitel 3 ein Produkt auf Bedienungsanleitungen vorschlagen, das uns in Kapitel 4 helfen wird, die Austauschbarkeitsfrage bezüglich des in der Einleitung erwähnten parametrisierten Austauschbarkeitsbegriffs, mit dem wir uns in dieser Arbeit beschäftigen, zu beantworten.

Vorher machen wir uns in diesem Kapitel jedoch mit den begrifflichen Grundlagen vertraut, indem wir alle für uns relevanten Definitionen und Sätze aus [LMW06] entnehmen, gegebenenfalls leicht an unsere Zwecke anpassen und zum Verständnis an Beispielen erläutern.

2.1 Serviceautomaten

Wie wir bereits erwähnt haben, modellieren wir einen Service als einen endlichen, nichtdeterministischen Serviceautomaten. Später werden wir Serviceautomaten miteinander kommunzieren lassen. Zu diesem Zweck besitzt jeder Serviceautomat eine aus Ein- und Ausgabekanälen bestehende Schnittstelle, über die er Nachrichten empfangen bzw. senden kann. Von einer Nachricht interessiert uns nur ihr Typ, der dadurch festgelegt ist, über welchen Kanal sie gesendet oder empfangen wird. In der gesamten Arbeit nehmen wir zudem an, dass es eine endliche Menge C solcher Nachrichtenkanäle gibt, die alle Kanäle enthält, die wir je benötigen werden.

Definition 1 (Serviceautomat)
Ein *Serviceautomat* $A = (Q, I, O, \delta, q_0, F)$ besteht aus

- einer Menge Q von Zuständen;
- einer Menge $I \subseteq C$ von Eingabe- (Input-) und einer Menge $O \subseteq C$ von Ausgabe- (Output-)Kanälen mit $I \cap O = \emptyset$, wobei C die oben angenommene endliche Menge von Nachrichtenkanälen ist;
- einer nichtdeterministischen Transitionsrelation $\delta \subseteq Q \times (I \cup O \cup \{\tau\}) \times Q$;
- einem Anfangszustand $q_0 \in Q$ und
- einer Menge $F \subseteq Q$ von Endzuständen, wobei für alle $q \in F$ gelten muss, dass $(q, x, q') \in \delta$ stets $x \in I$ impliziert.

Der Serviceautomat A ist *endlich*, wenn seine Zustandsmenge Q endlich ist. ⌟

Die Menge $I \cup O$ der Ein- und Ausgabekanäle eines Serviceautomaten bezeichnen wir als seine *Schnittstelle*. Über sie kann er mit anderen Serviceautomaten kommunizieren. Für einen Kanal $x \in I$ werden wir ab sofort $?x$, für einen Kanal $x \in O$ werden wir $!x$ schreiben.

Die Abbildung 2.1 zeigt einen Serviceautomaten P, der in seinem Anfangszustand `p0` eine Nachricht vom Typ `z` erwartet. Nach dem Empfang einer solchen Nachricht ist P im Zustand `p1`, in dem er ein `a` oder `c` senden kann. Je nach dem, für welche Nachricht er sich nichtdeterministisch entscheidet, wechselt er in den Zustand `p2` oder `p3`. Während er von `p2` durch Empfangen von `x` zurück in den

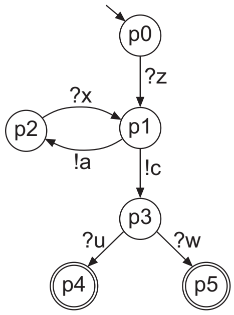


Abbildung 2.1: Ein Serviceautomat P, dessen Schnittstelle aus $I_P = \{\mathsf{u}, \mathsf{w}, \mathsf{x}, \mathsf{z}\}$ und $O_P = \{\mathsf{a}, \mathsf{c}\}$ bestehe.

Zustand p1 wechselt, kann er im Zustand p3 ein u oder ein w empfangen und dabei abhängig davon, welche Nachricht er in diesem Zustand von seiner Umgebung erhält, in einen seiner doppelt umrandeten Endzustände p4 oder p5 wechseln.

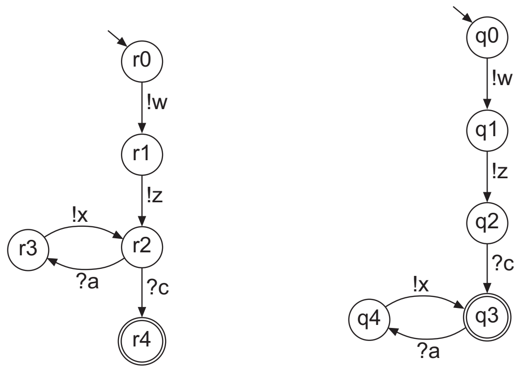
Ein Service ist dazu gedacht, mit Menschen oder anderen Services zu interagieren. Dazu tauschen er und sein Benutzer Nachrichten miteinander aus. Später werden wir definieren, wann wir solch eine Interaktion als sinnvoll betrachten.

Damit Services überhaupt sinnvoll miteinander interagieren können, müssen sie bzw. ihre Kommunikationsschnittstellen zusammen passen. Zwei Services passen zusammen, wenn jeder der beiden nur Nachrichtentypen senden kann, die der andere auch empfangen kann. Zwei auf diese Weise zueinander passende Services nennen wir *Partner*.

Definition 2 (Partner)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei endliche Serviceautomaten. P und R sind *Partner*, wenn $I_P = O_R$ und $I_R = O_P$ ist. ⌟

Die in Abbildung 2.2 dargestellten Serviceautomaten R und Q mit den Schnittstellen $I_R = I_Q = \{\mathsf{a}, \mathsf{c}\}$ und $O_R = O_Q = \{\mathsf{u}, \mathsf{w}, \mathsf{x}, \mathsf{z}\}$ sind Partner des Serviceautomaten P aus Abbildung 2.1, weil ihre Schnittstellen jeweils zu der von P passen. Dazu ist es nicht nötig, dass die Transitionsrelationen von R oder Q alle Nachrichtenkanäle ihrer Schnittstellen verwenden. Zum Beispiel besitzt R keine Transition, die ein u sendet, obwohl $\mathsf{u} \in O_R$ ist.

In unserem Modell werden Serviceautomaten asynchron miteinander kommunizieren. Das bedeutet unter anderem, dass Nachrichten beliebig lange zwischen zwei Services unterwegs sein und sich mehrere Nachrichten zugleich in einem Kanal befinden können. Um dies zu modellieren, verwenden wir Multimengen über der Menge C aller Nachrichtentypen.



(a) R (b) Q

Abbildung 2.2: R und Q sind zwei Partner des Serviceautomaten P aus Abbildung 2.1. Ihre Schnittstellen sollen aus $I_R = I_Q = \{\mathsf{a}, \mathsf{c}\}$ und $O_R = O_Q = \{\mathsf{u}, \mathsf{w}, \mathsf{x}, \mathsf{z}\}$ bestehen. Die Transitionsrelation eines Serviceautomaten muss nicht alle Nachrichtenkanäle seiner Schnittstelle verwenden.

Definition 3 (Multimengen)
Eine *Multimenge* M über einem Universum U ist eine totale Abbildung $M : U \to \mathbb{N}$. Für ein $u \in U$ heißt $M(u)$ die *Vielfachheit* von u in M. $bags(U)$ ist die *Menge aller Multimengen über dem Universum* U. $bags_k(U) = \{\, m \in bags(U) \mid \forall x \in U : m(x) \leq k \,\}$ ist die *Menge aller* k*-beschränkten Multimengen über* U. $[\,]$ bezeichnet die *leere Multimenge*, wobei für alle $u \in U$ gilt: $[\,](u) = 0$. Wir schreiben $a \in M$, falls $M(a) \geq 1$, und $a \notin M$, falls $M(a) = 0$ ist.

Sei $a \in U$. Dann stehen $M + a$ und $M - a$ für das Erhöhen bzw. Verringern der Vielfachheit von a in M um 1. Das heißt, $M{+}a$ und $M{-}a$ sind zwei Multimengen, für die für alle $u \in U$ gilt:

$$(M + a)(u) = \begin{cases} M(u) + 1 & \text{falls } u = a, \\ M(u) & \text{sonst} \end{cases}$$

$$(M - a)(u) = \begin{cases} M(u) - 1 & \text{falls } u = a \text{ und } M(a) \geq 1, \\ M(u) & \text{sonst.} \end{cases}$$

⌟

In unserem Modell lassen wir zwei Partner miteinander interagieren, indem wir ihre *Komposition* bilden. Die Komposition zweier Partner ist dabei ein Serviceautomat mit leerem Interface, also einer, der weder Nachrichten empfangen, noch senden kann.

In jedem Zustand der Komposition ist neben den aktuellen Zuständen der beteiligten Partner auch die Anzahl der Nachrichten verzeichnet, die sich gerade in den Nachrichtenkanälen zwischen den Partnern befinden, also schon gesendet, aber noch nicht empfangen wurden. Eine Transition in der Komposition entspricht dann genau einer Transition eines der beteiligten Partner. Dabei kann es sich um eine interne, eine Sende- oder eine Empfangstransition handeln, wobei eine Empfangstransition eines Partners in einem Zustand der Komposition nur eintreten kann, wenn sich die zu empfangende Nachricht im Kanal zwischen den beteiligten Serviceautomaten befindet, wohingegen das Eintreten von internen und Sendetransitionen nicht vom Zustand der Nachrichtenkanäle abhängt.

Definition 4 (Komposition von Partnern)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner. Ihre *Komposition* ist der Serviceautomat $P \oplus R = (Q, I, O, \delta, q_0, F)$ mit

- $I = O = \emptyset$,
- $q_0 = (q_{0P}, q_{0R}, [\,])$ und
- $F = F_P \times F_R \times \{[\,]\}$, wobei

Q und δ induktiv wie folgt definiert sind:
Induktionsanfang: $q_0 \in Q$, $\delta = \emptyset$.
Induktionsschritt: Sei $(q_P, q_R, m) \in Q$.

- Wenn $(q_P, \tau, q'_P) \in \delta_P$ ist, dann ist $((q_P, q_R, m), \tau, (q'_P, q_R, m)) \in \delta$.
  (interner Schritt in P)
- Wenn $(q_R, \tau, q'_R) \in \delta_R$ ist, dann ist $((q_P, q_R, m), \tau, (q_P, q'_R, m)) \in \delta$.
  (interner Schritt in R)
- Wenn $(q_P, x, q'_P) \in \delta_P$, $x \in I_P$ und $m(x) > 0$ ist,
  dann ist $((q_P, q_R, m), \tau, (q'_P, q_R, m - [x])) \in \delta$.
  (P empfängt x)
- Wenn $(q_R, x, q'_R) \in \delta_R$, $x \in I_R$ und $m(x) > 0$ ist,
  dann ist $((q_P, q_R, m), \tau, (q_P, q'_R, m - [x])) \in \delta$.
  (R empfängt x)
- Wenn $(q_P, x, q'_P) \in \delta_P$ und $x \in O_P$ ist,
  dann ist $((q_P, q_R, m), \tau, (q'_P, q_R, m + [x])) \in \delta$.
  (P sendet x)
- Wenn $(q_R, x, q'_R) \in \delta_R$ und $x \in O_R$ ist,
  dann ist $((q_P, q_R, m), \tau, (q_P, q'_R, m + [x])) \in \delta$.
  (R sendet x). ⌟

In Abbildung 2.3 sehen wir die Kompositionen des Serviceautomaten P aus Abbildung 2.1 mit seinen Partnern R und Q aus Abbildung 2.2. So beginnt zum Beispiel $P \oplus R$ im Zustand $(\mathsf{p0}, \mathsf{r0}, [\,])$, also in den Anfangszuständen von P und R bei leeren Nachrichtenkanälen. Von seinem Anfangszustand r0 aus kann R ein w senden, wobei R in den Zustand r1, $P \oplus R$ also in den Zustand (p0, r1, [w]) übergeht. Aus diesem Zustand heraus kann R ein z senden, wodurch $P \oplus R$ in den Zustand (p0, r2, [w, z]) gelangt. In diesem Zustand kann nun P einen Schritt ausführen und das von R gesendete z empfangen, womit $P \oplus R$ den Zustand (p1, r2, [w]) erreicht. Nach diesem Prinzip baut sich auch der Rest von $P \oplus R$ auf. Der in der Abbildung doppelt umrandete Zustand (p5, r4, [ ]) ist der einzige Endzustand von $P \oplus R$. In ihm sind P und R jeweils in einem ihrer Endzustände und alle Nachrichtenkanäle leer.

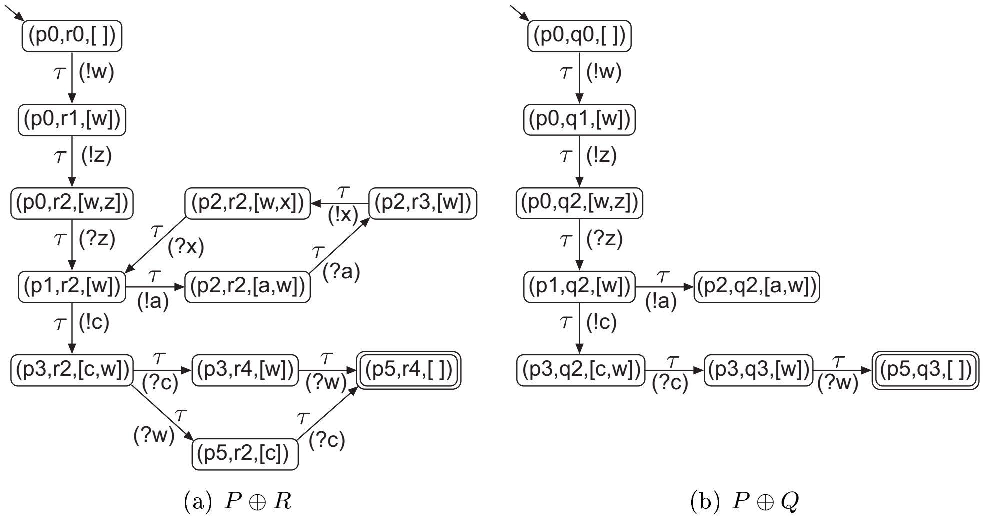


Abbildung 2.3: Die Kompositionen $P \oplus R$ und $P \oplus Q$ der Serviceautomaten P, R und Q aus den Abbildungen 2.1 und 2.2. Die Kanten sind mit τ beschriftet, weil die entsprechenden Zustandsübergänge in der Komposition interne Schritte sind. Daneben steht in Klammern jeweils die Beschriftung des dazugehörigen ursprünglichen Übergangs in einem der Partner.

Damit wir zwei Partner als *sinnvoll* miteinander interagierend bezeichnen können, müssen zwei Mindestanforderungen erfüllt sein. Erstens müssen am Ende ihrer Interaktion alle Nachrichten, die gesendet wurden, auch empfangen worden sein. Zweitens dürfen die Partner in keinen Deadlock geraten können. Das heißt, sie dürfen in keinen Nichtendzustand gelangen können, in dem sie keinen Schritt ausführen können, zum Beispiel, weil sie gegenseitig auf Nachrichten voneinander warten.

Die erste Anforderung haben wir bereits in der Definition der Komposition zweier Partner formuliert, indem wir nur solche Zustände als Endzustände der Komposition bezeichnen, in denen sich beide Partner in einem Endzustand befinden und alle Nachrichtenkanäle leer sind.

Zur Formalisierung der zweiten Anforderung definieren wir zunächst den Begriff *Deadlock*.

Definition 5 (Wartezustand, Deadlock)
Sei $A = (Q, I, O, \delta, q_0, F)$ ein Serviceautomat. Ein Zustand $q \in Q$ heißt genau dann *Wartezustand von A*, wenn aus $(q, x, q') \in \delta$ stets $x \in I$ folgt, das heißt, A den Zustand q nur mit Hilfe der Umgebung verlassen kann. Für einen Wartezustand q von A ist $wait(q) = \{ x \in C \mid \exists q' \in Q : (q, x, q') \in \delta \}$ die Menge der Nachrichten, die der Serviceautomat A im Wartezustand q empfangen kann. Ein Wartezustand q von A heißt genau dann *Deadlock von A*, wenn $q \notin F$ und $wait(q) = \emptyset$ ist. Besitzt A keinen Deadlock, nennen wir A *deadlockfrei*. ⌟

Die Komposition $P \oplus R$ in Abbildung 2.3(a) ist deadlockfrei. Zwar ist $(\mathsf{p5}, \mathsf{r4}, [\,])$ ein Wartezustand von $P \oplus R$, da die Implikation in der Definition des Begriffs Wartezustand wegen fehlender ausgehender Transitionen von $(\mathsf{p5}, \mathsf{r4}, [\,])$ trivialerweise erfüllt ist, und es gilt auch $wait(\mathsf{p5}, \mathsf{r4}, [\,]) = \emptyset$. Jedoch ist $(\mathsf{p5}, \mathsf{r4}, [\,])$ auch ein Endzustand von $P \oplus R$ und damit kein Deadlock.

Im Gegensatz dazu besitzt die Komposition $P \oplus Q$ in Abbildung 2.3(b) mit $(\mathsf{p2}, \mathsf{q2}, [\mathsf{a}, \mathsf{w}])$ einen Deadlock, da weder P noch Q aus diesem Zustand heraus einen Schritt ausführen kann und $(\mathsf{p2}, \mathsf{q2}, [\mathsf{a}, \mathsf{w}])$ auch kein Endzustand von $P \oplus Q$ ist.

In dieser Arbeit betrachten wir nur beschränkte Kompositionen von Partnern. Denn obwohl zwei Partner endliche Serviceautomaten sind, ist es wegen Zyklen in den Partnern möglich, dass ihre Komposition unendlich viele Zustände enthält. Für solche Partner werden wir später keine Bedienungsanleitungen definieren können. Deshalb beschäftigen wir uns in dieser Arbeit nur mit Partnern, deren Komposition beschränkt ist – die Forderung nach Endlichkeit der Komposition genügt nicht. Das heißt, dass wir von der Komposition zweier Partner fordern, dass sie nur Zustände besitzt, in denen höchstens k Nachrichten desselben Typs gleichzeitig in den Nachrichtenkanälen vorhanden sind, wobei k eine beliebige, aber feste natürliche Zahl ist.

Definition 6 (k-Beschränktheit, k-beschränkter Partner)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner und $P \oplus R = (Q, I, O, \delta, q_0, F)$ ihre Komposition. Wenn $Q \subseteq Q_P \times Q_R \times bags_k(C)$ ist, dann ist $P \oplus R$ *k-beschränkt* und R ein *k-beschränkter* Partner von P. ⌟

Unter der Menge der k-beschränkten Partner eines Serviceautomaten P zeichnen wir diejenigen aus, die mit P gemäß der oben genannten Mindestanforderungen sinnvoll interagieren. Einen Partner, der mit P in keinen Deadlock geraten kann, nennen wir *Strategie* von P. Die Forderung nach k-Beschränktheit pflanzt sich dabei vom Begriff des Partners auf den der Strategie fort.

Definition 7 (k-beschränkte Strategie)
Sei P ein Serviceautomat und R ein k-beschränkter Partner von P. Wenn $P \oplus R$ deadlockfrei ist, dann heißt R eine *k-beschränkte Strategie von P*. Die Menge aller deadlockfreien k-beschränkten Strategien von P heißt $\mathit{Strat}_k(P)$. ⌟

Da die Komposition $P \oplus R$ in Abbildung 2.3(a) 1-beschränkt und deadlockfrei ist, ist R eine 1-beschränkte Strategie von P. Dagegen ist Q *keine* Strategie von P, da die Komposition $P \oplus Q$ in Abbildung 2.3(b) mit dem Zustand $(\mathsf{p2}, \mathsf{q2}, [\mathsf{a}, \mathsf{w}])$ einen Deadlock besitzt.

In den nächsten Abschnitten werden wir uns näher mit k-beschränkten Strategien beschäftigen und nach einer Charakterierung aller k-beschränkten Strategien eines gegebenen Serviceautomaten suchen.

2.2 Situationen und Wissen

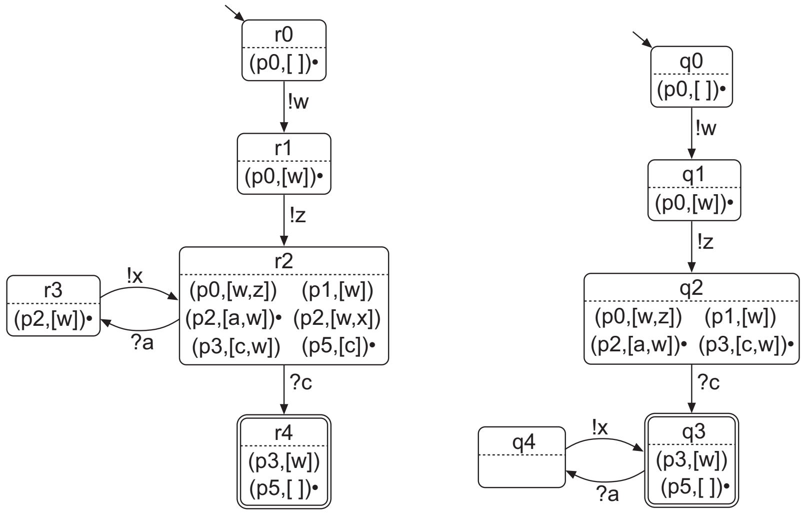
Um später die Bedienungsanleitung eines Serviceautomaten definieren zu können, führen wir zunächst die Begriffe Situation und Knowledge ein. Wenn zwei Serviceautomaten P und R miteinander interagieren und R die interne Struktur von P kennt, dann kann R anhand des eigenen Zustandes und der Nachrichten, die er an P gesendet oder von P empfangen hat, ableiten, in welchen Zuständen sich P und die Nachrichtenkanäle gerade befinden können.

Dieses Wissen von R formalisieren wir in den Begriffen Situation und Knowledge. Wir sprechen davon, dass in einem gegebenen Zustand von R mehrere Situationen herrschen können. Eine Situation umfasst dabei den Zustand von P und den Zustand der Nachrichtenkanäle, also wie viele welcher Nachrichten die Kanäle enthalten. Das Wissen (Knowledge) von R fassen wir formal als eine Abbildung K auf, die jedem seiner Zustände die Menge aller in ihm möglichen Situationen zuordnet.

Definition 8 (K, Situation)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner und $P \oplus R = (Q, I, O, \delta, q_0, F)$ ihre Komposition. Dann ist die Abbildung $K : Q_R \to \wp(Q_P \times \mathit{bags}(C))$ definiert durch $K(q_R) = \{\, (q_P, m) \mid (q_P, q_R, m) \in Q \,\}$. Ein Element von $Q_P \times \mathit{bags}(C)$ heißt *Situation*.

Die Abbildung K nennen wir auch *Wissens-* oder *Knowledgefunktion*. ⌟

Abbildung 2.4 zeigt die in Abbildung 2.2 dargestellten Partner R und Q von P aus Abbildung 2.1 mit ihrem Wissen über P. Wenn sich R in der Komposition $P \oplus R$ in seinem Anfangszustand $\mathsf{r0}$ befindet, dann muss sich P auch immer noch in seinem Anfangszustand $\mathsf{p0}$ befinden, und alle Nachrichtenkanäle müssen leer sein, da P das von ihm erwartete z erst empfangen kann, nachdem R es gesendet hat. Deshalb ist $K(\mathsf{r0}) = \{(\mathsf{p0}, [\,])\}$. Nachdem R den Zustand r0 durch Senden von w verlassen hat und in den Zustand r1 übergegangen ist, kann P immer noch keinen Schritt ausführen. P muss sich also immer noch im Zustand p0 befinden, wobei nun jedoch das von R gesendete w im Nachrichtenkanal liegt. Deshalb ist $K(\mathsf{r1}) = \{(\mathsf{p0}, [\mathsf{w}])\}$. Erst nachdem R durch seinen Übergang in den Zustand r2 ein z gesendet hat, kann P mehrere Schritte ausführen: P kann zuerst das z empfangen und dann entweder ein a oder ein c senden. Falls P durch seinen Übergang in den Zustand p3 ein c sendet, kann er anschließend sofort das von R gesendete w empfangen. Deshalb umfasst $K(\mathsf{r2})$ viele Situationen, nämlich $K(\mathsf{r2}) = \{(\mathsf{p0}, [\mathsf{w}, \mathsf{z}]), (\mathsf{p1}, [\mathsf{w}]), (\mathsf{p2}, [\mathsf{a}, \mathsf{w}]), (\mathsf{p2}, [\mathsf{w}, \mathsf{x}]), (\mathsf{p3}, [\mathsf{c}, \mathsf{w}]), (\mathsf{p5}, [\mathsf{c}])\}$.



(a) R mit Wissen über P (b) Q mit Wissen über P

Abbildung 2.4: Die Serviceautomaten R und Q aus Abbildung 2.2 mit ihrem Wissen über P aus Abbildung 2.1.

Analog ergibt sich die Knowledgefunktion für den Serviceautomaten Q (siehe Abbildung 2.4(b)). Dort ist $K(\mathsf{q4})$ leer, weil in der Komposition $P \oplus Q$ kein Zustand erreichbar ist, in dem sich Q im Zustand q4 befindet (siehe Abbildung 2.3(b)).

Für die Konstruktion von Bedienungsanleitungen müssen wir zwischen transienten und stabilen Situationen unterscheiden. In einer Situationsmenge M nennen wir eine Situation $(q, m) \in M$ *transient*, wenn der Serviceautomat, zu dem der Zustand q gehört, von (q, m) aus ohne Hilfe der Umgebung eine Situation (q', m') erreichen kann, die auch in M enthalten ist. Andernfalls nennen wir (q, m) *stabil*.

Definition 9 (transiente und stabile Situationen)
Sei $P = (Q, I, O, \delta, q_0, F)$ ein Serviceautomat und $M \subseteq Q \times \mathit{bags}(C)$ eine Menge von Situationen. Eine Situation $(q, m) \in M$ ist genau dann *transient in M bezüg-*

lich P, wenn ein $(q, x, q') \in \delta$ existiert, so dass eine der folgenden Bedingungen erfüllt ist:

- $x = \tau$ und $(q', m) \in M$,
- $x \in O$ und $(q', m + [x]) \in M$ oder
- $x \in I$, $m(x) > 0$ und $(q', m - [x]) \in M$.

Andernfalls ist (q, m) *stabil in* M *bezüglich* P. ⌟

In Abbildung 2.4 sind die bezüglich P stabilen Situationen in den dargestellten Situationsmengen mit einem • markiert; alle anderen Situationen sind transient. Zum Beispiel ist $(\mathsf{p2}, [\mathsf{a}, \mathsf{w}])$ stabil in $K(\mathsf{r2})$, weil in dieser Situation kein x im Nachrichtenkanal liegt, aber P den Zustand $\mathsf{p2}$ nur durch den Empfang eines x verlassen kann. Folglich ist P auf die Hilfe von R, der ein x senden kann, angewiesen, um die Situation $(\mathsf{p2}, [\mathsf{a}, \mathsf{w}])$ verlassen zu können.

Mit den bis hierhin eingeführten Begriffen können wir eine Charakterisierung für Deadlocks in Kompositionen von Serviceautomaten angeben.

Lemma 1 ([LMW06])
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner und $P \oplus R = (Q, I, O, \delta, q_0, F)$ ihre Komposition. So ist ein Zustand $(q_P, q_R, m) \in Q$ genau dann Deadlock von $P \oplus R$, wenn alle folgenden Bedingungen gelten:

- $q_P \notin F_P$, $q_R \notin F_R$ oder $m \neq [\,]$;
- q_R ist ein Wartezustand von R; und
- (q_P, m) ist stabil in $K(q_R)$ bezüglich P, und für alle $x \in \mathit{wait}(q_R)$ gilt $m(x) = 0$. ⌟

Als nächstes stellen wir fest, dass sich die Bedingungen aus Lemma 1 als boolesche Formeln über der Menge C aller Nachrichtenkanäle zusammen mit den speziellen Propositionen τ und *final* kodieren lassen. Indem wir jeden Zustand q von R mit einer solchen für ihn konstruierten Formel beschriften, können wir anschließend nur anhand der Formel und der ausgehenden Transitionen von q entscheiden, ob q ein Deadlock von $P \oplus R$ ist. Dies wird wichtig werden, wenn wir entscheiden wollen, welche Menge von Serviceautomaten durch eine gegebene Bedienungsanleitung charakterisiert wird.

Definition 10 (Beschriftung ϕ, R-Belegung)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei

Partner. Dann ist für jeden Zustand $q_R \in Q_R$ die *Beschriftung* $\phi(q_R)$ wie folgt als boolesche Formel über den Propositionen $C \cup \{\tau, \mathit{final}\}$ definiert:

$$\phi(q_R) = \bigwedge_{\substack{(q_P,m) \text{ stabil} \\ \text{in } K(q_R) \\ \text{bzgl. } P}} (\phi_1(q_P, m) \vee \phi_2 \vee \phi_3(m)),$$

wobei

$$\phi_1(q_P, m) = \begin{cases} \mathit{final}, & \text{falls } q_P \in F_P \text{ und } m = [\,] \\ \mathit{false}, & \text{sonst} \end{cases}$$

$$\phi_2 = \tau \vee \bigvee_{x \in O_R} x$$

$$\phi_3(m) = \bigvee_{\substack{x \in I_R, \\ m(x) > 0}} x.$$

Die *R-Belegung* $ass_R(q_R)$ belegt alle Propositionen $x \in C \cup \{\tau\}$ genau dann mit *true*, wenn ein q'_R existiert, so dass $(q_R, x, q'_R) \in \delta_R$ ist, und belegt *final* genau dann mit *true*, wenn $q_R \in F_R$ ist. ⌟

Nachdem wir die Bedingungen aus Lemma 1 als boolesche Formeln kodiert haben, können wir Lemma 1 so umformulieren, dass wir nur noch anhand der Knotenbeschriftung ϕ und der R-Belegung ass_R entscheiden können, ob $P \oplus R$ deadlockfrei ist.

Korollar 1 ([LMW06])
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner und $P \oplus R$ ihre Komposition. $P \oplus R$ ist genau dann deadlockfrei, wenn für alle $q_R \in Q_R$ der Wert von $\phi(q_R)$ unter der R-Belegung $ass_R(q_R)$ *true* ist. ⌟

Diese Eigenschaft wird sich als essentiell für die Definition von Bedienungsanleitungen herausstellen. Denn indem wir die noch zu definierende k-beschränkte Strategie von P mit den oben beschriebenen booleschen Formeln beschriften, werden wir all diejenigen Serviceautomaten charakterisieren können, die mit P deadlockfrei interagieren.

2.3 Operationen auf Mengen von Situationen

Für die Konstruktion der schon erwähnten k-beschränkten Strategie eines Serviceautomaten, die die Grundlage für die Konstruktion seiner Bedienungsanleitung sein wird, benötigen wir zwei Operationen auf Mengen von Situationen. Zum einen definieren wir, zu welcher Menge von Situationen ein Serviceautomat durch den Eintritt eines Sende-, Empfangs- oder internen Ereignisses in einer Menge von Situationen gelangt. Zum anderen, was es heißt, die Hülle einer Menge von Situationen zu bilden.

Als erstes definieren wir *Sende-, Empfangs-* und *interne Ereignisse*. Damit formalisieren wir den Effekt eines Ereignisses auf eine Menge von Situationen. Gegeben sei ein Serviceautomat P und ein Partner R, eine Menge von Situationen M über den Zuständen von P und ein Nachrichtenkanal x. Dann gibt das Sendeereignis $\mathit{send}(M, x)$ zum Beispiel an, in welcher Menge von Situationen sich P, unmittelbar nachdem ihm R aus der Situationsmenge M heraus ein x gesendet hat, befinden kann. Im Falle dieses Sendeereignisses ergibt sich die neue Situationsmenge aus der alten M, indem in jeder Situation aus M eine Nachricht x zum Nachrichtenkanal hinzugefügt wird. Empfangs- und interne Ereignisse sind analog definiert, mit dem Unterschied, dass R bei einem Empfangsereignis eine Nachricht empfängt und bei einem internen Ereignis einen internen Schritt ausführt und keine Nachricht sendet oder empfängt.

Definition 11 (Sende-, Empfangs-, internes Ereignis)
Seien $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ und $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ zwei Partner und $M \subseteq Q_P \times \mathit{bags}(C)$. Wenn $x \in O_R$ ist, dann ist das *Sendeereignis* x, geschrieben $\mathit{send}(M, x)$, definiert durch $\mathit{send}(M, x) = \{\, (q, m + [x]) \mid (q, m) \in M \,\}$. Wenn $x \in I_R$ ist, dann ist das *Empfangsereignis* x, $\mathit{receive}(M, x)$, definiert durch $\mathit{receive}(M, x) = \{\, (q, m - [x]) \mid (q, m) \in M, m(x) > 0 \,\}$. Das *interne Ereignis* τ, $\mathit{internal}(M, \tau)$, ist definiert durch $\mathit{internal}(M, \tau) = M$.

Da die Art eines Ereignisses aus I_R und O_R ablesbar ist, definieren wir das *Ereignis* x, geschrieben $\mathit{event}(M, x)$, durch

$$\mathit{event}(M, x) = \begin{cases} \mathit{receive}(M, x) & \text{falls } x \in I_R, \\ \mathit{send}(M, x) & \text{falls } x \in O_R, \\ \mathit{internal}(M, x) & \text{falls } x = \tau. \end{cases}$$

⌟

Betrachten wir zum Beispiel den Serviceautomaten R aus Abbildung 2.4(a) als Partner von P aus Abbildung 2.1 und die Situationsmenge $K(\mathsf{r1}) = \{(\mathsf{p0}, [\mathsf{w}])\}$. Da R aus dem Zustand $\mathsf{r1}$ heraus durch den Übergang in Zustand $\mathsf{r2}$ ein z senden kann, betrachten wir das Sendeereignis $\mathit{send}(K(\mathsf{r1}), \mathsf{z})$ und stellen fest, dass $\mathit{send}(K(\mathsf{r1}), \mathsf{z}) = \{(\mathsf{p0}, [\mathsf{w}, \mathsf{z}])\}$ ist. Das Sendeereignis $\mathit{send}(K(\mathsf{r1}), \mathsf{z})$ liefert genau diejenige Situationsmenge, die direkt nach dem Senden von z aus $K(\mathsf{r1})$ heraus herrscht. Wir beobachten außerdem, dass $\mathit{send}(K(\mathsf{r1}), \mathsf{z}) \subseteq K(\mathsf{r2})$ ist.

Weiterhin kann R aus dem Zustand $\mathsf{r2}$ heraus ein a empfangen und dadurch in den Zustand $\mathsf{r3}$ übergehen. Wenn wir deswegen die Situationsmenge $K(\mathsf{r2}) = \{(\mathsf{p0}, [\mathsf{w}, \mathsf{z}]), (\mathsf{p1}, [\mathsf{w}]), (\mathsf{p2}, [\mathsf{a}, \mathsf{w}]), (\mathsf{p2}, [\mathsf{w}, \mathsf{x}]), (\mathsf{p3}, [\mathsf{c}, \mathsf{w}]), (\mathsf{p5}, [\mathsf{c}])\}$ und das Ereignis $\mathit{receive}(K(\mathsf{r2}), \mathsf{a})$ betrachten, stellen wir fest, dass $\mathit{receive}(K(\mathsf{r2}), \mathsf{a}) = \{(\mathsf{p2}, [\mathsf{w}])\}$

ist. Denn $(\mathsf{p2}, [\mathsf{a}, \mathsf{w}])$ ist die einzige Situation in $K(\mathsf{r2})$, die ein a im Nachrichtenkanal enthält, und wenn wir in dieser Situation ein a empfangen, so gelangen wir in die Situation $(\mathsf{p2}, [\mathsf{w}])$. Analog zum vorigen Absatz beobachten wir, dass $\mathit{receive}(K(\mathsf{r2}), \mathsf{a}) \subseteq K(\mathsf{r3})$ gilt.

Wir sehen, dass wir mit Sende-, Empfangs- und internen Ereignissen allein, die aus Abbildung 2.4 abzulesenden Übergänge zwischen den Situationsmengen noch nicht ganz beschreiben können. Denn $\mathit{send}(K(\mathsf{r1}), \mathsf{z})$ zum Beispiel liefert, wie oben gezeigt, nur eine Teilmenge von $K(\mathsf{r2})$, nicht jedoch $K(\mathsf{r2})$ selbst.

Deshalb definieren wir als weitere Operation die Hülle einer Situationsmenge. Wenn P ein Serviceautomat und M eine Menge von Situationen über den Zuständen von P ist, dann enthält die *Hülle* von M bezüglich P all diejenigen Situationen, die P ohne Einfluss seiner Umgebung von den Situationen in M aus erreichen kann.

Definition 12 (Hülle)
Sei $P = (Q, I, O, \delta, q_0, F)$ ein Serviceautomat und $M \subseteq Q \times \mathit{bags}(C)$ eine Menge von Situationen. Dann ist die *Hülle von M, $cl(M)$, bezüglich* P induktiv wie folgt definiert:

Induktionsanfang: $M \subseteq cl(M)$.
Induktionsschritt: Sei $(q, m) \in cl(M)$ und $(q, x, q') \in \delta$. Dann ist

- $(q', m) \in cl(M)$, falls $x = \tau$,
- $(q', m + [x]) \in cl(M)$, falls $x \in O$,
- $(q', m - [x]) \in cl(M)$, falls $x \in I$ und $m(x) > 0$. ⌟

Betrachten wir zum Beispiel den Serviceautomaten P aus Abbildung 2.1 und die Menge $M = \{(\mathsf{p0}, [\mathsf{w}, \mathsf{z}])\}$ von Situationen. Dann kann P aus dieser Situationsmenge heraus ein z empfangen und in den Zustand $\mathsf{p1}$ übergehen, danach ein a oder c senden und in den Zustand $\mathsf{p2}$ bzw. $\mathsf{p3}$ übergehen, und von $\mathsf{p3}$ aus ein w empfangen und im Zustand $\mathsf{p5}$ enden; und das alles ohne die Hilfe eines Partners. Also ist $cl(M) = \{(\mathsf{p0}, [\mathsf{w}, \mathsf{z}]), (\mathsf{p1}, [\mathsf{w}]), (\mathsf{p2}, [\mathsf{a}, \mathsf{w}]), (\mathsf{p3}, [\mathsf{c}, \mathsf{w}]), (\mathsf{p5}, [\mathsf{c}])\}$.

Weiter oben haben wir festgestellt, dass $\mathit{send}(K(\mathsf{r1}), \mathsf{z})$ nur eine Teilmenge von $K(\mathsf{r2})$, nicht jedoch $K(\mathsf{r2})$ selbst liefert (siehe Abbildung 2.4(a)). Diesen Mangel können wir mit der Hüllenoperation beheben. Denn $cl(\mathit{send}(K(\mathsf{r1}), \mathsf{z})) = K(\mathsf{r2})$. Damit haben wir den Übergang in R vom Zustand $\mathsf{r1}$ in den Zustand $\mathsf{r2}$ nur durch die oben eingeführten Operationen auf den entsprechenden Situationsmengen beschrieben.

Allerdings lassen sich nicht alle Zustandsübergänge von R auf diese Weise beschreiben. Es gilt zum Beispiel *nicht* $cl(\mathit{send}(K(\mathsf{r3}), \mathsf{x})) = K(\mathsf{r2})$. Diesen Umstand werden wir am Ende des nächsten Abschnitts noch einmal näher betrachten.

Wir werden das hier dargestellte Prinzip, von einer Situationsmenge durch den Eintritt eines Ereignisses und das Bilden der Hülle in eine andere Situationsmenge überzugehen, jedoch bei der Konstruktion der im nächsten Abschnitt definierten k-beschränkten Strategie verwenden. In ihr wird *jeder* Zustandsübergang auf die oben dargestellte Art beschrieben sein.

2.4 Kanonische k-beschränkte Strategie

Die Bildung der kanonischen k-beschränkten Strategie eines Serviceautomaten P ist ein entscheidender Schritt in der Konstruktion der Bedienungsanleitung von P. Die kanonische k-beschränkte Strategie von P ist zum einen eine k-beschränkte Strategie von P. Zum anderen werden wir sie in Abschnitt 2.5 mit den in Abschnitt 2.2 eingeführten booleschen Formeln beschriften und dadurch eine Bedienungsanleitung von P erhalten, die alle k-beschränkten Strategien von P charakterisiert.

Definition 13 (Kanonische k-beschränkte Strategie S_k)
Sei $P = (Q_P, I_P, O_P, \delta_P, q_{0P}, F_P)$ ein Serviceautomat. Dann ist die *kanonische k-beschränkte Strategie* $S_k = (Q, I, O, \delta, q_0, F)$ *von* P ein Serviceautomat mit:

- $I = O_P$,
- $O = I_P$,
- $F = \{\, q \in Q \mid q \text{ ist Wartezustand von } S_k \,\}$,
- $q_0 = cl(\{(q_{0P}, [\,])\})$, wobei

Q und δ induktiv wie folgt definiert sind:
Induktionsanfang: $q_0 \in Q$ und $\delta = \emptyset$.
Induktionsschritt: Sei $x \in C$. Wenn $q \in Q$ und $q' = cl(event(q, x)) \subseteq Q_P \times bags_k(C)$ ist, dann ist $q' \in Q$ und $(q, x, q') \in \delta$. ⌟

Falls für ihren Anfangszustand $q_0 = cl(\{(q_{0P}, [\,])\}) \subseteq Q_P \times bags_k(C)$ gilt, nennen wir S_k *wohldefiniert*. Dann gilt für alle $q \in Q$: $K(q) = q$. Darüber hinaus ist $P \oplus S_k$ dann k-beschränkt und deadlockfrei, S_k also eine k-beschränkte Strategie von P.

Wenn S_k nicht wohldefiniert ist, dann hat P keinen k-beschränkten Partner und damit auch keine k-beschränkte Strategie.

Abbildung 2.5 zeigt die kanonische 1-beschränkte Strategie des Serviceautomaten P aus Abbildung 2.1. In der Abbildung wurden zur besseren Übersicht mehrere Kanten zwischen denselben Zuständen zu einer Kante zusammengefasst und kompakter beschriftet. So steht die Kantenbeschriftung ?* für alle möglichen Empfangsereignisse. Statt der einen mit ?* beschrifteten Kante zwischen r0 und r1 gibt es tatsächlich also zwei Kanten, eine mit `?a` und eine mit `?c` beschriftete. Analog steht die Kantenbeschriftung `*` für alle möglichen Sende- und Empfangsereignisse.

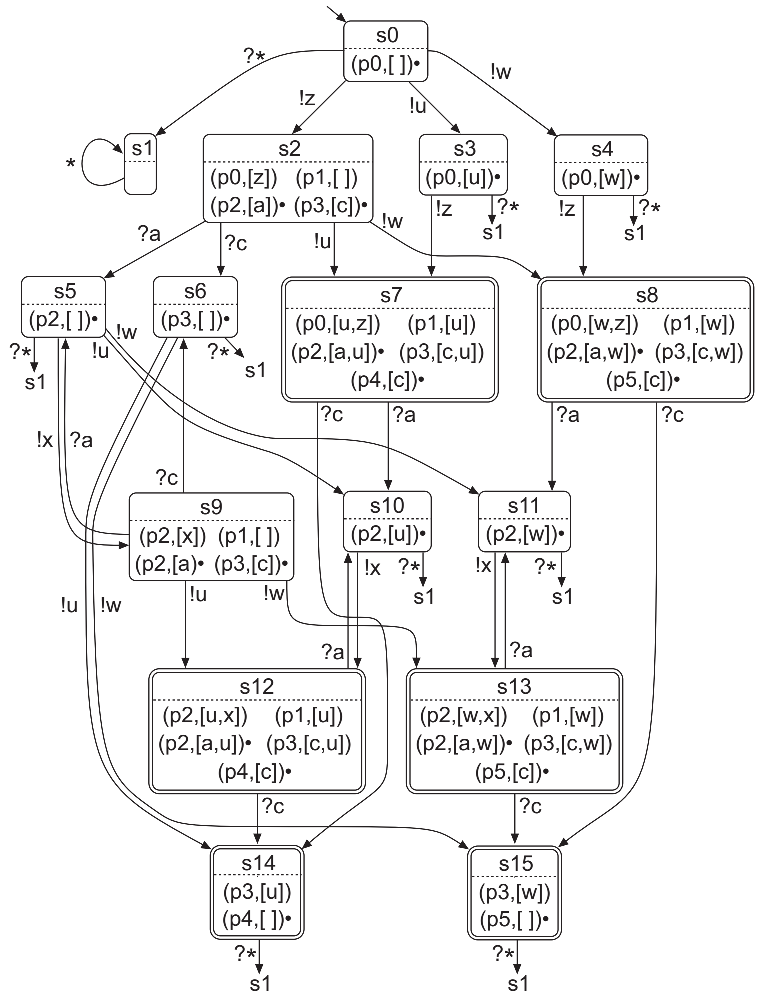


Abbildung 2.5: Die kanonische 1-beschränkte Strategie von P aus Abbildung 2.1.

In jeder kanonischen k-beschränkten Strategie hat jeder Zustand eine auf sich selbst zeigende mit τ beschriftete Kante. Diese Kanten wurden in der Abbildung aus Platzgründen ebenfalls weggelassen.

Auch weggelassen wurden Zustände, die nur deswegen in Strategien von P vorkommen dürfen, weil sie Livelocks sind. Zum Beispiel ist der Serviceautomat T aus Abbildung 2.6(a) auch eine Strategie von P aus Abbildung 2.1. T sendet von seinem Anfangszustand `t0` aus ein `x`, geht dabei in den Zustand `t1` über und kann von dort aus unendlich viele interne Schritte zurück zu `t1` ausführen. Auf diese Weise gelangt die Komposition $P \oplus T$ zwar nie in einen Deadlock, weswegen T eine

1-beschränkte Strategie von P ist. Allerdings kann $P \oplus T$ vom Zustand $(\mathsf{p0}, \mathsf{t1}, [\,])$ aus unendlich oft τ-Schritte zurück zu diesem Zustand ausführen. Deshalb nennen wir $(\mathsf{p0}, \mathsf{t1}, [\,])$ einen *Livelock*. Livelocks sind üblicherweise unerwünscht.

(a) T (b) U

Abbildung 2.6: Zwei Livelocks produzierende Serviceautomaten T und U.

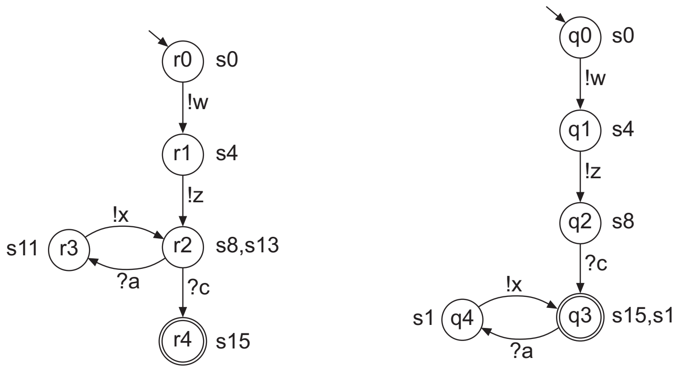
Genauso könnte die kanonische 1-beschränkte Strategie von P aus Abbildung 2.5 von jedem dargestellten Zustand aus eine Nachricht an P senden und dann in einem Livelock bleiben; sie wäre trotzdem eine 1-beschränkte Strategie von P. Solche Zustände gehören der Definition nach auch zur kanonischen Strategie. In der Abbildung haben wir sie allerdings aus Platzgründen weggelassen, auch weil solche Zustände, wie bereits erwähnt, in der Regel unerwünscht sind.

Wegen der angesprochenen Livelocks hat jeder Serviceautomat, der einen k-beschränkten Partner besitzt, das heißt, dessen kanonische k-beschränkte Strategie wohldefiniert ist, auch den in Abbildung 2.6(b) dargestellten Serviceautomaten U als k-beschränkte Strategie, der von seinem Anfangszustand $\mathsf{u0}$ aus nur τ-Schritte zurück zu $\mathsf{u0}$ ausführen kann.

Im nächsten Abschnitt 2.5 werden wir die kanonische k-beschränkte Strategie S_k von P beschriften, um eine Bedienungsanleitung von P zu erhalten. Um danach entscheiden zu können, ob ein beliebiger Serviceautomat R der Bedienungsanleitung von P genügt und somit eine Strategie von P ist, müssen wir R und die Bedienungsanleitung von P unter anderem in die nachfolgend definierte Matching-Relation setzen.

Definition 14 (Matching)
Seien $R_1 = (Q_1, I_1, O_1, \delta_1, q_{01}, F_1)$ und $R_2 = (Q_2, I_2, O_2, \delta_2, q_{02}, F_2)$ zwei Serviceautomaten. Dann ist das *Matching von R_1 und R_2* als Relation $L_{R_1,R_2} \subseteq Q_1 \times Q_2$ induktiv wie folgt definiert: Sei $(q_{01}, q_{02}) \in L_{R_1,R_2}$. Wenn $(q_1, q_2) \in L_{R_1,R_2}$, $(q_1, x, q_1') \in \delta_1$ und $(q_2, x, q_2') \in \delta_2$ ist, dann ist auch $(q_1', q_2') \in L_{R_1,R_2}$. ⌟

Abbildung 2.7 zeigt die Serviceautomaten R und Q aus Abbildung 2.2 und ihr Matching mit der kanonischen 1-beschränkten Strategie S_k von P aus Abbildung 2.5. Neben jedem Zustand q von R bzw. Q sind die Zustände von S_k aufgeführt, mit denen q in der Matching-Relation steht. So matcht der Zustand $\mathsf{r2}$ aus R zum Beispiel mit den Zuständen $\mathsf{s8}$ und $\mathsf{s13}$ aus S_k.



(a) R und sein Matching mit P

(b) Q und sein Matching mit P

Abbildung 2.7: R und Q aus Abbildung 2.2 und ihr Matching mit P aus Abbildung 2.1.

In vorigen Abschnitt 2.3, in dem wir den Effekt von Ereignissen auf und die Hülle von Situationsmengen definiert haben, haben wir festgestellt, dass im Serviceautomaten R (siehe Abbildung 2.4(a)) die Situationsmenge $cl(event(K(\mathsf{r3}), x))$ nicht gleich $K(\mathsf{r2})$, sondern nur eine Teilmenge von $K(\mathsf{r2})$ ist. Das hängt damit zusammen, dass der Zustand r2 mit mehr als einem Zustand aus S_k matcht: s8 und s13. Es gilt nämlich $cl(event(K(\mathsf{r3}), x)) = K(\mathsf{s8}) \cup K(\mathsf{s13}))$. Allgemein ist dies in Lemma 2 formuliert.

Lemma 2 ([LMW06])
Sei P ein Serviceautomat, S_k seine kanonische k-beschränkte Strategie, $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ ein k-beschränkter Partner von P und L_{R,S_k} das Matching von R und S_k. Dann gilt für alle $q_R \in Q_R$: $K(q_R) = \bigcup_{\substack{(q_R,q_S)\\ \in L_{R,S_k}}} K(q_S)$. ⌟

2.5 Bedienungsanleitungen

Wir haben nun alle Begriffe zusammen, um die Bedienungsanleitung eines gegebenen Serviceautomaten P definieren zu können. Sie wird alle Strategien von P charakterisieren. Folglich werden wir anhand der Bedienungsanleitung von P entscheiden können, ob ein gegebener Serviceautomat R eine Strategie von P ist. Umgekehrt kann man die Bedienungsanleitung von P auch verwenden, um aus ihr gezielt eine Strategie von P zu konstruieren.

Um die Bedienungsanleitung für einen Serviceautomaten P zu konstruieren, bilden wir zuerst seine kanonische k-beschränkte Strategie S_k und beschriften diese dann mit der Beschriftungsfunktion ϕ aus Definition 10. Jeder zu diesem beschrifteten Serviceautomaten isomorphe Serviceautomat ist dann eine *Bedienungsanleitung von P*.

Definition 15 (Bedienungsanleitung)
Sei P ein Serviceautomat, der mindestens einen Partner R hat, so dass $P \oplus R$ deadlockfrei ist. Dann heißt jeder Serviceautomat S_k^*, der isomorph zur kanonischen k-beschränkten Strategie S_k von P ist, zusammen mit einer Abbildung Φ mit $\Phi(q_{S_k^*}) = \phi(q_{S_k})$ für jedes $q_{S_k^*}$, das isomorph zu q_{S_k} ist, eine *k-beschränkte Bedienungsanleitung für P*. ⌟

Eine Bedienungsanleitung $S_k^{*\Phi}$ von P werden wir fortan mit OG_P bezeichnen, wobei sich die Abkürzung OG von dem englischen Begriff *operating guideline* für Bedienungsanleitung ableitet.

Als nächstes definieren wir die Menge $Comply(OG_P)$ aller durch die Bedienungsanleitung OG_P charakterisierten Serviceautomaten.

Definition 16 (*Comply*)
Sei A^Φ ein beschrifteter Serviceautomat, R ein Serviceautomat und $L_{R,A}$ ihr Matching. So gilt genau dann $R \in Comply(A^\Phi)$, wenn für alle $(q_R, q_A) \in L_{R,A}$ folgende Bedingungen erfüllt sind:

- *Topologie:* Für jedes $x \in C \cup \{\tau\}$ gilt: Wenn eine x-Transition existiert, die q_R in R verlässt, dann existiert auch eine x-Transition, die q_A in A verlässt.
- *Beschriftung:* Die Belegung $ass_R(q_R)$ erfüllt die Beschriftung $\Phi(q_A)$. ⌟

Im Fall $R \in Comply(A^\Phi)$ sprechen wir auch davon, dass R dem beschrifteten Serviceautomaten A^Φ *genügt*.

Wir stellen fest, dass die Menge $Comply(OG_P)$ aller von OG_P charakterisierten Serviceautomaten genau die Menge der k-beschränkten Strategien von P ist.

Satz 1 (Charakterisierung von Strategien [LMW06])
Sei P ein Serviceautomaten und OG_P eine k-beschränkte Bedienungsanleitung von P. Dann gilt $Strat_k(P) = Comply(OG_P)$. ⌟

Das heißt, ein Serviceautomat R ist genau dann eine k-beschränkte Strategie von P, wenn $R \in Comply(OG_P)$ ist. Damit ist gerechtfertigt, dass wir OG_P eine Bedienungsanleitung von P nennen.

Abbildung 2.8 zeigt eine 1-beschränkte Bedienungsanleitung OG_P von P. Sie wurde aus der kanonischen 1-beschränkten Strategie S_k von P aus Abbildung 2.5 durch Beschriftung mit ϕ aus Definition 10 gewonnen. Demnach ergibt sich die boolesche Formel, mit der ein Zustand von OG_P beschriftet ist, aus seiner in

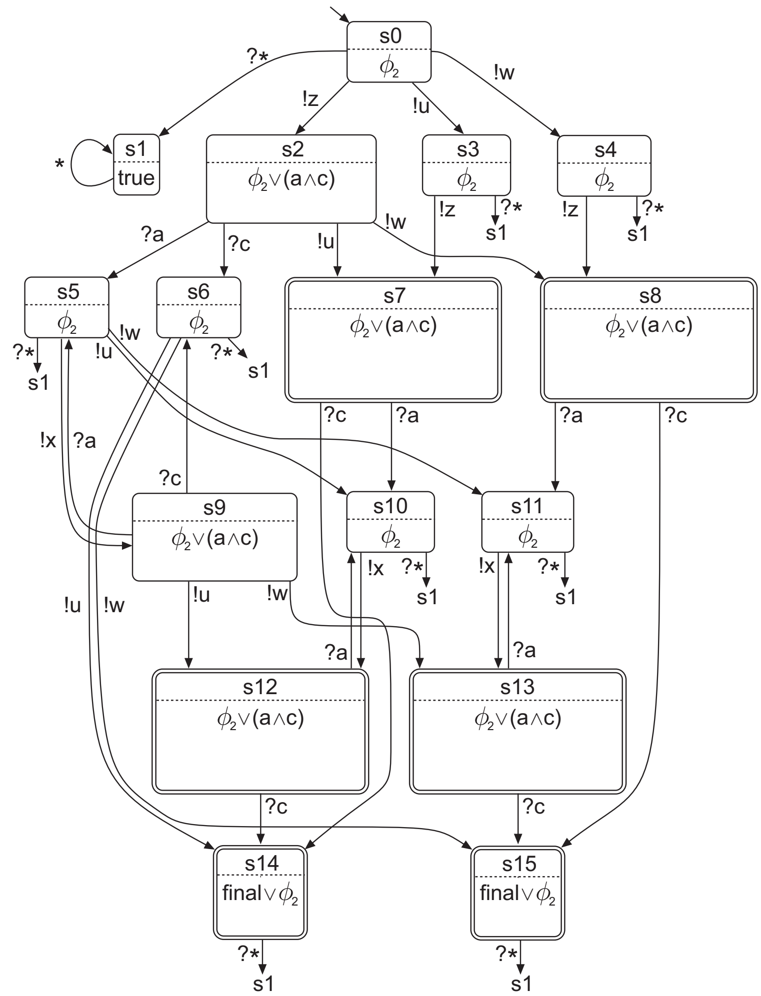


Abbildung 2.8: Eine 1-beschränkte Bedienungsanleitung OG_P von P; entstanden durch Beschriftung seiner 1-beschränkten kanonischen Strategie aus Abbildung 2.5. Dabei steht ϕ_2 abkürzend für $\tau \vee \mathsf{u} \vee \mathsf{w} \vee \mathsf{x} \vee \mathsf{z}$.

Abbildung 2.5 dargestellten Situationsmenge $K(q)$ in S_k. Zur Darstellung wurden die Formeln weitestgehend vereinfacht.

Um zu entscheiden, ob der Serviceautomat R der Bedienungsanleitung OG_P von P genügt, betrachten wir ihn und sein Matching mit OG_P in der Abbildung 2.7(a). Wir stellen fest, dass jeder Zustand von R sowohl die in Definition 16 geforderte Topologie-, als auch die Beschriftungseigenschaft erfüllt. Zum Beispiel weist die Belegung $ass_R(\mathsf{r0})$ dem Literal w den Wert *true* zu und erfüllt damit die Beschriftung $\phi(\mathsf{s0}) = \tau \vee \mathsf{u} \vee \mathsf{w} \vee \mathsf{x} \vee \mathsf{z}$. Folglich genügt R der Bedienungsanleitung OG_P, das heißt, $R \in \mathit{Comply}(OG_P)$, und ist damit eine 1-beschränkte Strategie von P. Dies deckt sich mit unserer Beobachtung in Abschnitt 2.1, dass die in Abbildung 2.3(a) dargestellte Komposition $P \oplus R$ deadlockfrei und 1-beschränkt ist.

Im Gegensatz dazu genügt der Serviceautomat Q aus Abbildung 2.2(b) bzw. 2.7(b) der Bedienungsanleitung OG_P *nicht* ($Q \notin \mathit{Comply}(OG_P)$), weil keine ?a-Kante den Zustand q2 in Q verlässt, obwohl das von der Beschriftung des Knotens s8 in OG_P gefordert wird. Folglich ist Q, wie wir schon in Abschnitt 2.1 beobachtet haben, keine 1-beschränkte Strategie von P.

In späteren Darstellungen von Bedienungsanleitungen werden wir nicht nur τ-Schleifen und Livelocks weglassen, sondern auch überflüssige Literale aus den booleschen Formeln, mit denen die Zustände beschriftet sind, streichen und die Formeln so weit wie möglich vereinfachen. In der Beschriftung eines Zustands q ist ein Literal l dann überflüssig, wenn es keine mit l beschriftete Kante gibt, die q verlässt.

3 Produktbedienungsanleitungen

In Kapitel 2 haben wir die für dieses Kapitel grundlegenden Begriffe definiert und ihre Zusammenhänge erläutert. Wir wissen, was Serviceautomaten, Strategien und Bedienungsanleitungen sind.

In diesem Kapitel werden wir ein Produkt von Bedienungsanleitungen, genauer gesagt von beliebigen beschrifteten Serviceautomaten, vorschlagen. Dieses Produkt wird uns helfen, das in Kapitel 4 gestellte Austauschbarkeitsproblem zu lösen.

Wir wissen, dass jeder beschriftete Serviceautomat S^{Φ} eine Menge $\mathit{Comply}(S^{\Phi})$ von Serviceautomaten charakterisiert. Das Produkt zweier beschrifteter Serviceautomaten $S_1^{\Phi_1}$ und $S_2^{\Phi_2}$ werden wir so definieren, dass es gerade die Schnittmenge $\mathit{Comply}(S_1^{\Phi_1}) \cap \mathit{Comply}(S_2^{\Phi_2})$ charakterisiert. Damit verhält sich das Produkt von beschrifteten Serviceautomaten analog zu dem traditioneller endlicher Automaten [HU79]. Denn das Produkt zweier endlicher Automaten A_1 und A_2, deren reguläre Sprachen $L(A_1)$ bzw. $L(A_2)$ sind, ist ein endlicher Automat, dessen Sprache gerade die Schnittmenge $L(A_1) \cap L(A_2)$ ist.

Später werden wir das Produkt zwischen beschrifteten Serviceautomaten auf Bedienungsanleitungen anwenden. Gegeben zwei Serviceautomaten P_1 und P_2 mit ihren Bedienungsanleitungen OG_{P_1} bzw. OG_{P_2}, die alle Strategien von P_1 bzw. P_2 charakterisieren, wird das Produkt von OG_{P_1} und OG_{P_2} den Durchschnitt aller Strategien von P_1 und P_2 charakterisieren, also genau jene Serviceautomaten, die Strategie von P_1 *und* P_2 sind.

Da das Produkt zwischen beschrifteten Serviceautomaten kommutativ und assoziativ sein wird, werden wir die Bedienungsanleitung einer endlichen Menge $\mathcal{S}$ von Serviceautomaten definieren können, die all diejenigen Serviceautomaten charakerisiert, die Strategien für alle Serviceautomaten aus $\mathcal{S}$ sind.

Mit Hilfe solch einer Produktbedienungsanleitung werden wir in Kapitel 4 diejenigen Serviceautomaten charakterisieren können, gegen die sich ein gegebener Serviceautomat P bezüglich einer endlichen Menge von Strategien von P austauschen lässt.

Um das Produkt zweier beschrifteter Serviceautomaten S_1 und S_2 bilden zu können, müssen ihre Schnittstellen identisch sein. Dann nennen wir S_1 und S_2 *schnittstellengleich*.

Definition 17 (schnittstellengleich)
Seien $S_1 = (Q_1, I_1, O_1, \delta_1, q_{01}, F_1)$, $S_2 = (Q_2, I_2, O_2, \delta_2, q_{02}, F_2)$ zwei Serviceautomaten. S_1 und S_2 heißen *schnittstellengleich*, wenn $I_1 = I_2$ und $O_1 = O_2$ ist. ⌟

Das Produkt zweier beschrifteter Serviceautomaten ist ein wie folgt definierter Serviceautomat.

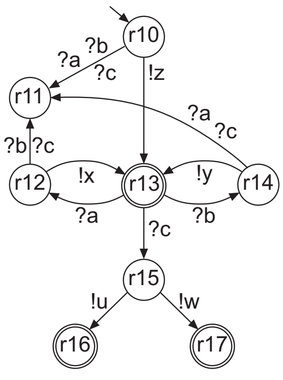
Definition 18 (Produkt zwischen beschrifteten Serviceautomaten, $\otimes$)
Seien $S_1^{\Phi_1} = (Q_1, I_1, O_1, \delta_1, q_{01}, F_1)^{\Phi_1}$, $S_2^{\Phi_2} = (Q_2, I_2, O_2, \delta_2, q_{02}, F_2)^{\Phi_2}$ zwei beschriftete schnittstellengleiche Serviceautomaten. Dann ist ihr *Produkt* $S_1^{\Phi_1} \otimes S_2^{\Phi_2}$ definiert als der beschriftete Serviceautomat $S_1^{\Phi_1} \otimes S_2^{\Phi_2} = S^{\Phi} = (Q, I, O, \delta, q_0, F)$ mit

- $Q = L_{S_1,S_2}$, wobei L_{S_1,S_2} das Matching von S_1 und S_2 ist;
- $O = O_1 = O_2$
- $I = I_1 = I_2$
- $((q_1, q_2), x, (q_1', q_2')) \in \delta$ gdw. $(q_1, x, q_1') \in \delta_1$ und $(q_2, x, q_2') \in \delta_2$;
- $q_0 = (q_{01}, q_{02})$;
- $F = \{\, q \in Q \mid q$ ist Wartezustand von $S \,\}$; und
- $\Phi((q_1, q_2)) = \Phi_1(q_1) \wedge \Phi_2(q_2)$ für alle $(q_1, q_2) \in Q$. ⌟

Im Folgenden setzen wir, wenn wir das Produkt von beschrifteten Serviceautomaten bilden, stets Schnittstellengleichheit voraus.

Betrachten wir zum Beispiel die in Abbildung 3.1 dargestellten Serviceautomaten R_1 und R_2. R_1 geht von seinem Anfangszustand r10 durch Senden eines z in einen seiner Endzustände r13 über. In diesem Zustand kann er weitere Nachrichten empfangen. Empfängt er ein a oder ein b, quittiert er die Nachricht durch Senden eines x bzw. y und geht zurück in den Zustand r13. Empfängt er im Zustand r13 ein c, antwortet er nach seiner nichtdeterministischen Entscheidung entweder mit einem u oder einem w und geht dabei in einen seiner Endzustände r16 bzw. r17 über. Von dort aus kann er keine weiteren Nachrichten empfangen.

Wenn R_1 in den Zuständen r10, r12 oder r14 eine Nachricht empfängt, die nicht dem beschriebenen Protokoll entspricht, kann er in den Zustand r11 übergehen, der als Fehlerzustand aufzufassen ist, weil R_1 von dort aus keinen seiner Endzustände mehr erreichen kann. Folglich kann ein livelockfreier Serviceautomat A, der so mit R_1 kommuniziert, dass R_1 in den Zustand r11 gelangen kann, keine Strategie von R_1 sein. Wir haben R_1 mit dem Fehlerzustand r11 versehen, um die Strategien von R_1 in ihrer Wahl, wann sie welche Nachricht senden, einzuschränken, damit die Bedienungsanleitung von R_1 für das weitere Vorgehen übersichtlich bleibt.

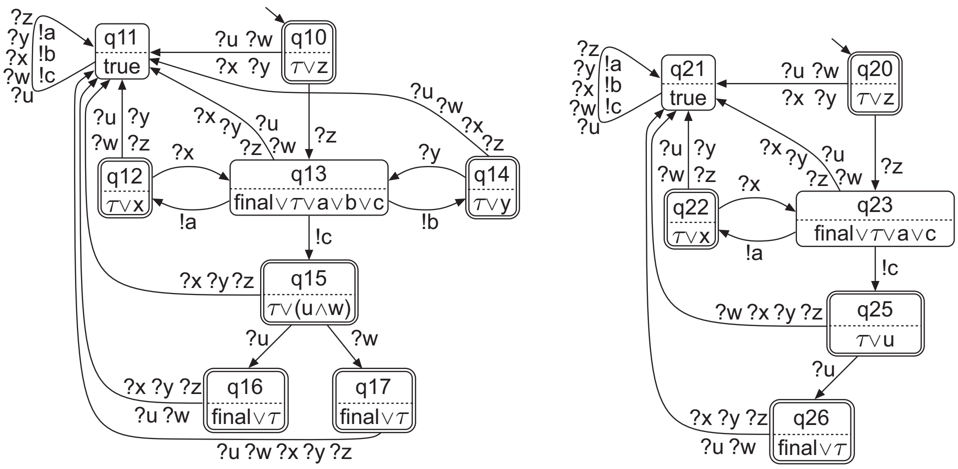


(a) R_1 (b) R_2

Abbildung 3.1: Die Serviceautomaten R_1 und R_2.

Der Serviceautomat R_2 ähnelt R_1. Im Vergleich zu R_1 fehlt ihm nur die Möglichkeit, nach dem Senden des z ein b zu empfangen und durch ein y zu quittieren.

Die Bedienungsanleitungen OG_{R_1} und OG_{R_2} von R_1 bzw. R_2 sind in Abbildung 3.3 dargestellt. Sie wurden für die Darstellung auf gewohnte Weise vereinfacht: τ-Schleifen, Livelocks und überflüssige Literale in den booleschen Formeln wurden weggelassen.



(a) OG_{R_1} (b) OG_{R_2}

Abbildung 3.2: Die 1-beschränkten Bedienungsanleitungen von R_1 und R_2.

Ihr Produkt $OG_{R_1} \otimes OG_{R_2}$ ist in Abbildung 3.3 dargestellt. Seine Zustände sind Paare der Zustände von OG_{R_1} und OG_{R_2}. Das Produkt $OG_{R_1} \otimes OG_{R_2}$ enthält nur solche Kanten, die in OG_{R_1} *und* OG_{R_2} vorhanden sind. So führt in $OG_{R_1} \otimes OG_{R_2}$ eine ?w-Kante vom Zustand (q15, q25) zu (q17, q21), weil in OG_{R_1} eine ?w-Kante vom Zustand q15 zu q17 und in OG_{R_2} eine von q25 nach q21 führt. Im Gegensatz dazu verlässt in $OG_{R_1} \otimes OG_{R_2}$ keine !b-Kante den Zustand (q13, q23), weil in R_2 keine !b-Kante den Zustand q23 verlässt. Die Zustandsbeschriftungen in $OG_{R_1} \otimes OG_{R_2}$ ergeben sich aus der Konjunktion der entsprechenden Zustandsbeschriftungen in R_1 und R_2. Da der Zustand q17 in OG_{R_1} mit final $\vee \tau$ und q21 in OG_{R_2} mit true beschriftet ist, ist der Zustand (q17, q21) in $OG_{R_1} \otimes OG_{R_2}$ mit (final $\vee \tau) \wedge$ true beschriftet, was wir in der Darstellung zu final $\vee \tau$ vereinfacht haben.

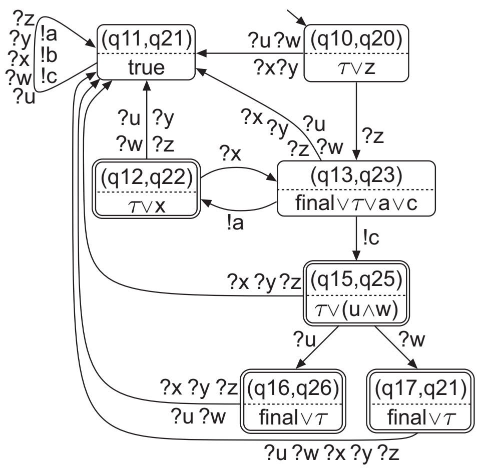


Abbildung 3.3: Die Produktbedienungsanleitung $OG_{R1} \otimes OG_{R2}$.

Die wichtigste Eigenschaft des Produkts zweier beschrifteter Serviceautomaten $S_1^{\Phi_1}$ und $S_2^{\Phi_2}$ besteht darin, dass es genau den Durchschnitt der durch $S_1^{\Phi_1}$ und $S_2^{\Phi_2}$ charakterisierten Serviceautomaten charakterisiert. Dies ist in Satz 2 formuliert. Für seinen Beweis verwenden wir Lemma 3.

Lemma 3
Nehmen wir an, $R = (Q_R, I_R, O_R, \delta_R, q_{0R}, F_R)$ sei ein Serviceautomat, $S_1^{\Phi_1} = (Q_1, I_1, O_1, \delta_1, q_{01}, F_1)^{\Phi_1}$ und $S_2^{\Phi_2} = (Q_2, I_2, O_2, \delta_2, q_{02}, F_2)^{\Phi_2}$ zwei beschriftete Serviceautomaten, $S^{\Phi} = (Q, I, O, \delta, q_0, F)^{\Phi}$ ihr Produkt, L_{R,S_1} das Matching von R und S_1 und L_{R,S_2} das Matching von R und S_2. Dann gilt für alle $q_R \in Q_R$, $q_1 \in Q_{S_1}$ und $q_2 \in Q_{S_2}$: $(q_R, (q_1, q_2)) \in L_{R,S}$ genau dann, wenn $(q_R, q_1) \in L_{R,S_1}$ und $(q_R, q_2) \in L_{R,S_2}$. ⌟

Beweis 1
Es gilt

$(q_R, (q_1, q_2)) \in L_{R,S}$
gdw. es eine Sequenze x von Kantenbeschriftungen gibt, so dass man
von q_{0R} über δ_R gemäß x den Zustand q_R und
von (q_{01}, q_{02}) über δ gemäß x den Zustand (q_1, q_2) erreicht,
gdw. es eine Sequenze x von Kantenbeschriftungen gibt, so dass man
von q_{0R} über δ_R gemäß x den Zustand q_R,
von q_{01} über δ_1 gemäß x den Zustand q_1 und
von q_{02} über δ_2 gemäß x den Zustand q_2 erreicht,
gdw. $(q_R, q_1) \in L_{R,S_1}$ und $(q_R, q_1) \in L_{R,S_1}$.

Dabei wenden wir im ersten Schritt die Definition des Matchings, im zweiten die des Produkts zwischen beschrifteten Serviceautomaten und im letzten Schritt abermals die Definition des Matchings an. □

Satz 2
Seien $S_1^{\Phi_1}$, $S_2^{\Phi_2}$ zwei beschriftete Serviceautomaten. Dann ist $\mathit{Comply}(S_1^{\Phi_1} \otimes S_2^{\Phi_2}) = \mathit{Comply}(S_1^{\Phi_1}) \cap \mathit{Comply}(S_2^{\Phi_2}))$. ⌟

Beweis 2
Sei $S^\Phi = (Q, I, O, \delta, q_0, F)^\Phi = S_1^{\Phi_1} \otimes S_2^{\Phi_2}$.

($\Rightarrow$) Sei $R \in \mathit{Comply}(S^\Phi)$. Sei $(q_R, q_{S_1}) \in L_{R,S_1}$ und $(q_R, q_{S_2}) \in L_{R,S_2}$. Wegen Lemma 3 ist dann $(q_R, (q_{S_1}, q_{S_2})) \in L_{R,S}$. Sei $x \in C \cup \{\tau\}$ und verlasse eine x-Transition q_R in R. Wegen $R \in \mathit{Comply}(S^\Phi)$ und der Topologieeigenschaft aus Definition 16 wissen wir, dass dann auch eine x-Transition (q_{S_1}, q_{S_2}) in S verlässt. Wegen der Bildungsvorschrift für δ in Definition 18 verlässt dann auch eine x-Transition q_{S_1} in S_1 und eine q_{S_2} in S_2. Also ist die Topologie-Eigenschaft für R bezüglich S_1 und bezüglich S_2 erfüllt.

Da wir wegen $R \in \mathit{Comply}(S^\Phi)$ und der Beschriftungseigenschaft wissen, dass die Belegung $\mathit{ass}_R(q_R)$ die Beschriftung $\Phi((q_{S_1}, q_{S_2}))$ erfüllt und wegen der Bildung von $\Phi((q_{S_1}, q_{S_2}))$ durch Konjunktion von $\Phi_1(q_{S_1})$ und $\Phi_2(q_{S_2})$ erfüllt $\mathit{ass}_R(q_R)$ sowohl $\Phi_1(q_{S_1})$ als auch $\Phi_2(q_{S_2})$. Damit gilt auch die Beschriftungseigenschaft für R bezüglich $S_1^{\Phi_1}$ und bezüglich $S_2^{\Phi_2}$.

Folglich gilt $R \in \mathit{Comply}(S_1^{\Phi_1}) \cap \mathit{Comply}(S_2^{\Phi_2})$, was zu zeigen war.

($\Leftarrow$) Sei $R \in \mathit{Comply}(S_1^{\Phi_1}) \cap \mathit{Comply}(S_2^{\Phi_2})$. Sei $(q_R, q) \in L_{R,S}$ mit $q = (q_1, q_2)$. Dann gilt mit Lemma 3 auch $(q_R, q_1) \in L_{R,S_1}$ und $(q_R, q_2) \in L_{R,S_2}$. Sei $x \in C \cup \{\tau\}$ und verlasse eine x-Transition q_R in R. Wegen $R \in \mathit{Comply}(S_1^{\Phi_1})$, $R \in \mathit{Comply}(S_1^{\Phi_1})$ und der Topologieeigenschaft in Definition 16 wissen wir, dass eine x-Transition auch q_1 in S_1 und eine q_2 in S_2 verlässt. Wegen der Bildungsvorschrift von δ in Defintion 18 verlässt damit auch eine x-Transition $(q_1, q_2) = q$ in S. Also gilt die Topologieeigenschaft für R bezüglich S.

Wegen $R \in \mathit{Comply}(S_1^{\Phi_1})$, $R \in \mathit{Comply}(S_2^{\Phi_2})$ und der Beschriftungseigenschaft in Definition 16 wissen wir, dass die Belegung $ass_R(q_R)$ die Beschriftung $\Phi_1(q_1)$ und die Beschriftung $\Phi_2(q_2)$ erfüllt. Also erfüllt sie auch die Beschriftung $\Phi_1(q_R) \wedge \Phi_2(q_R) = \Phi((q_1, q_2)) = \Phi(q_\otimes)$. Also gilt auch die Beschriftungseigenschaft für R bezüglich S^Φ.

Folglich gilt $R \in \mathit{Comply}(S^\Phi)$, was zu zeigen war. $\square$

Zum Beispiel genügt der in Abbildung 3.4 dargestellte Serviceautomat P_1 den Bedienungsanleitungen OG_{R_1} *und* OG_{R_2} aus Abbildung 3.2, das heißt, es ist $P_1 \in \mathit{Comply}(OG_{P_1})$ und $P_1 \in \mathit{Comply}(OG_{R_2})$. Damit genügt P_1 auch der Produktbedienungsanleitung $OG_{R_1} \otimes OG_{R_2}$ aus Abbildung 3.3, das heißt $P_1 \in \mathit{Comply}(OG_{R_1} \otimes OG_{R_2})$.

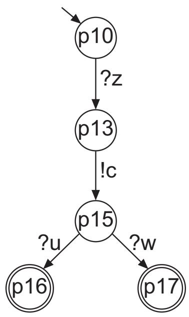


Abbildung 3.4: Der Serviceautomat P_1. Er genügt den Bedienungsanleitungen OG_{R_1} und OG_{R_2}. Folglich genügt er auch ihrem Produkt $OG_{R_1} \otimes OG_{R_2}$ und ist eine Strategie von R_1 und R_2.

Um Gegensatz dazu genügt der Serviceautomat P_2 aus Abbildung 3.5, der bereits der Bedienungsanleitung OG_{R_2} nicht genügt, auch nicht der Produktbedienungsanleitung $OG_{R_1} \otimes OG_{R_2}$, weil in P_2 eine !b-Kante den Zustand p23 verlässt, was beide Bedienungsanleitungen verbieten.

Lemma 4 (Kommutativität und Assoziativität von $\otimes$)
Das Produkt $\otimes$ zwischen beschrifteten Serviceautomaten ist kommutativ und assoziativ, das heißt, für alle beschrifteten Serviceautomaten $S_1^{\Phi_1}$, $S_2^{\Phi_2}$ und $S_3^{\Phi_3}$ gilt:

$$\mathit{Comply}(S_1^{\Phi_1} \otimes S_2^{\Phi_2}) = \mathit{Comply}(S_2^{\Phi_2} \otimes S_1^{\Phi_1}) \text{ und}$$
$$\mathit{Comply}((S_1^{\Phi_1} \otimes S_2^{\Phi_2}) \otimes S_3^{\Phi_3}) = \mathit{Comply}(S_1^{\Phi_1} \otimes (S_2^{\Phi_2} \otimes S_3^{\Phi_3})).$$

⌟

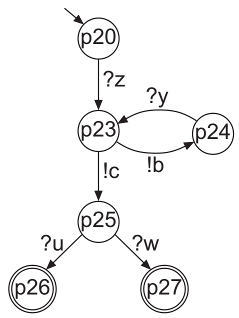


Abbildung 3.5: Der Serviceautomat P_2. Er genügt der Bedienungsanleitung OG_{R_2} nicht. Folglich genügt er auch nicht dem Produkt $OG_{R_1} \otimes OG_{R_2}$ und ist keine Strategie von R_2.

Beweis 3
Die Komponenten der Zustände in $S_1^{\Phi_1} \otimes S_2^{\Phi_2}$ sind nur vertauscht zu denen in $S_2^{\Phi_2} \otimes S_1^{\Phi_1}$ (siehe Definition des Produkt $\otimes$). Damit sind $S_1^{\Phi_1} \otimes S_2^{\Phi_2}$ und $S_2^{\Phi_2} \otimes S_1^{\Phi_1}$ isomorph zueinander. Weiterhin sind die Konjunktionsglieder der bei der Produktbildung entstehenden Konjunktionen in den Beschriftungen zueinander isomorpher Zustände von $S_1^{\Phi_1} \otimes S_2^{\Phi_2}$ und $S_2^{\Phi_2} \otimes S_1^{\Phi_1}$ nur vertauscht. Wegen der Kommutativität der Konjunktion sind die resultierenden Formeln folglich semantisch äquivalent. Deshalb charakterisieren $S_1^{\Phi_1} \otimes S_2^{\Phi_2}$ und $S_2^{\Phi_2} \otimes S_1^{\Phi_1}$ dieselben Mengen von Serviceautomaten (siehe Definition 16). Aus analogen Gründen gilt die Assoziativität von $\otimes$. $\square$

Wegen der Kommutativität und Assoziativität von $\otimes$ können wir das Produkt einer *Menge* beschrifteter Serviceautomaten definieren, welches wir in Kapitel 4 benötigen werden, wenn es um die Frage der Austauschbarkeit eines Service geht.

Definition 19 (Produkt einer Menge beschrifteter Serviceautomaten)
Sei $\mathcal{S} = \{S_1^{\Phi_1}, \ldots, S_n^{\Phi_n}\}$ eine Menge schnittstellengleicher beschrifteter Serviceautomaten. Dann ist das *Produkt* $\mathcal{S}_\otimes^\Phi$ *von* $\mathcal{S}$ definiert als $\mathcal{S}_\otimes^\Phi = S_1^{\Phi_1} \otimes \ldots \otimes S_n^{\Phi_n}$. ⌟

$\mathcal{S}_\otimes^\Phi$ ist wegen der Eindeutigkeit, der Kommutativität und der Assoziativität von $\otimes$ bis auf Isomorphie und semantische Äquivalenz der Zustandsbeschriftungen eindeutig definiert.

In Vorbereitung auf Kapitel 4 formulieren wir Lemma 5, das uns die Menge derjenigen Serviceautomaten charakterisieren lässt, die Strategien aller Serviceautomaten einer gegebenen Menge $\mathcal{R}$ von Serviceautomaten sind.

Lemma 5
Sei $\mathcal{R} = \{R_1, \ldots, R_n\}$ eine endliche Menge schnittstellengleicher Serviceautomaten und $\mathcal{S} = \{S_1^{\Phi_1}, \ldots, S_n^{\Phi_1}\}$ eine endliche Menge von Bedienungsanleitungen,

wobei jedes $S_i^{\Phi_i}$ eine k-beschränkte Bedienungsanleitung für R_i ist. Sei $\mathcal{S}_\otimes^\Phi$ das Produkt von $\mathcal{S}$. Dann gilt $\mathit{Comply}(\mathcal{S}_\otimes^\Phi) = \bigcap_{R \in \mathcal{R}} \mathit{Strat}_k(R)$. ⌟

Beweis 4
Seien $\mathcal{R}$, $\mathcal{S}$ und $\mathcal{S}_\otimes^\Phi$ wie im Lemma. Nach Voraussetzung ist jedes $S_i^{\Phi_i}$ eine k-beschränkte Bedienungsanleitung von R_i. Für alle $i \in \{1, \ldots, n\}$ gilt also $\mathit{Comply}(S_i^{\Phi_i}) = \mathit{Strat}_k(R_i)$. Außerdem ist $\mathcal{S}_\otimes^\Phi$ das Produkt von $\mathcal{S}$, das heißt,

$$\mathit{Comply}(\mathcal{S}_\otimes^\Phi) = \bigcap_{i=1}^{n} \mathit{Comply}(S_i^{\Phi_i})$$

(wegen Satz 2 und der Kommutativität und Assoziativität von $\otimes$).

Dann folgt unmittelbar:

$$\mathit{Comply}(\mathcal{S}_\otimes^\Phi) = \bigcap_{i=1}^{n} \mathit{Comply}(S_i^{\Phi_i}) = \bigcap_{i=1}^{n} \mathit{Strat}_k(R_i) = \bigcap_{R \in \mathcal{R}} \mathit{Strat}_k(R),$$

was zu zeigen war. □

Nachdem wir uns in diesem Kapitel ein geeignetes Produkt auf beschrifteten Serviceautomaten definiert haben, wenden wir es in den nächsten beiden Kapiteln in verschiedenen Situationen an. Dabei geht es in Kapitel 4 um die Austauschbarkeit eines gegebenen Service P unter Bewahrung einer gegebenen Menge $\mathcal{R}$ von Strategien von P. Anschließend beschäftigen wir uns in Kapitel 5 mit der Konstruktion der Bedienungsanleitung eines gegebenen Service P aus den Bedienungsanleitungen seiner Teilautomaten.

4 Austauschbarkeit von Services

Im vorigem Kapitel haben wir ein Produkt von beschrifteten Serviceautomaten vorgeschlagen und untersucht. Dieses Produkt wenden wir nun an, um verschiedene Problemstellungen zu lösen. Dabei wird es in diesem Kapitel um die Austauschbarkeit von Services und im nächsten um das Konstruieren von Bedienungsanleitungen aus Teilautomaten gehen.

Wir haben bereits erwähnt, dass Services für die automatische Geschäftsabwicklung zwischen Firmen immer wichtiger werden. Firmen bieten Services an und veröffentlichen ihre Bedienungsanleitungen, damit andere Firmen oder Services die angebotenen Services anhand ihrer Bedienungsanleitungen auffinden und richtig benutzen können.

Üblicherweise erwarten die Kunden, dass die angebotenen Services zuverlässig und vorhersagbar funktionieren und insbesondere, dass sie jederzeit so wie durch ihre Bedienungsanleitungen beschrieben verwendet werden können. Gerade bei automatisierten Geschäftsabwicklungen ist es nötig, dass diese Erwartungen der Kunden erfüllt werden.

Diese Erwartungen zu erfüllen, kann allerdings schwierig werden, wenn angebotene Services geändert werden müssen. Solche Änderungen können aus verschiedenen Gründen nötig werden: durch eine geänderte Gesetzeslage, durch geänderte Marktbedingungen oder eine Veränderung der Kundenwünsche.

Folglich ergibt sich die Frage, wie ein Service so geändert werden kann, dass jeder, der den ursprünglichen Service verwenden konnte auch den geänderten verwenden kann. Da wir uns in dieser Arbeit nicht mit dem Austausch von Services zur Laufzeit, sondern zur Entwurfszeit beschäftigen, können wir diese Frage wie folgt umformulieren: Gegeben einen ursprünglichen Serviceautomaten P_1 und einen neuen Serviceautomaten P_2: Ist dann jede Strategie von P_1 auch Strategie von P_2? Wenn ja, kann jeder, der den ursprünglichen Service P_1 verwenden konnte, auch den neuen Service P_2 verwenden. In diesem Fall nennen wir P_1 gegen P_2 *austauschbar*.

In der Literatur gibt es verschiedene Begriffe, die kommunizierende Prozesse bezüglich ihres Verhaltens in Relation stellen und deshalb als Ausgangspunkt zur Beantwortung der Austauschbarkeitsfrage sinnvoll erscheinen.

Als einen sehr wichtigen Begriff ist hier der der *Simulation* [Mil71] zu nennen. Ein Prozess P_2 simuliert einen Prozess P_1 intuitiv beschrieben, wenn P_2 sich mindestens so verhalten kann wie P_1. Dazu definiert Milner eine Simulationsrelation zwischen den Zuständen von P_1 und P_2. Ein Zustand q_2 von P_2 simuliert dann einen Zustand q_1 von P_1, wenn folgendes gilt: Wenn in P_1 ein mit a beschrifteter Übergang von q_1 nach q_1' möglich ist, dann muss in P_2 auch ein mit a beschrifteter Übergang von q_2 nach q_2' möglich sein und q_2' muss q_1' simulieren. Dann simuliert der Prozess P_2 den Prozess P_1, wenn der Anfangszustand von P_2 den Anfangszustand von P_1 simuliert.

Noch schärfer ist der Begriff der *Bisimulation* [Mil89, Par81]. Denn zwei Prozesse P_1 und P_2 sind bisimular, wenn es eine Relation R gibt, so dass P_1 mit R den Prozess P_2 und P_2 mit R^{-1} den Prozess P_1 simuliert.

Van Glabbeek gibt in [Gla01] eine Übersicht weiterer gebräuchlicher Äquivalenzbegriffe. Der schwächste darin ist die *Trace-Äquivalenz*, die zwei Prozesse als äquivalent bezeichnet, wenn sie dieselbe Menge sequentieller Abläufe generieren.

Auf Basis dieser Begriffe wurden in [AB02, KHA03, HDA$^+$05] verschiedene Äquivalenzbegriffe für Workflows und Workflow-Netze untersucht.

Doch keiner dieser Begriffe lässt sich bei genauerer Betrachtung zur Beantwortung der Austauschbarkeitsfrage von Services verwenden. Zwar herrscht die starke Vermutung, dass ein Serviceautomat P_1 gegen P_2 austauschbar ist, wenn P_2 den Serviceautomaten P_1 simuliert. Die Umkehrung gilt im Allgemeinen jedoch nicht. Das liegt unter anderem daran, dass Serviceautomaten asynchron miteinander kommunizieren und damit zum Beispiel die Reihenfolge unmittelbar aufeinanderfolgender Sendeereignisse irrelevant ist. So sind die in Abbildung 4.1 dargestellten Serviceautomaten A und B nicht einmal traceäquivalent und simulieren sich folglich auch nicht, weil A den Ablauf !a!b und B den Ablauf !b!a generiert. Beide haben aber dieselbe Menge von Strategien und sind damit gegeneinander austauschbar.

(a) A (b) B

Abbildung 4.1: Die Serviceautomaten A und B sind nicht traceäquivalent, haben aber dieselben Strategien und sind deshalb gegeneinander austauschbar.

Mit der *Bedienbarkeitsäquivalenz* definiert Martens in [Mar03] einen Begriff für offene Workflow-Netze, der dem oben genannten alle Strategien bewahrenden Austauschbarkeitsbegriff für Serviceautomaten entspricht. Auf Grundlage der Bedienbarkeitsäquivalenz beschäftigt sich Richter in [Ric02] mit der Frage, wann ein Service P_1 gegen einen Service P_2 ausgetauscht werden kann, wenn nur genau eine gegebene Strategie von P_1 bewahrt bleiben soll.

Wenn ein Unternehmen einen angebotenen Serivce P_1 gemäß des oben genannten allgemeinen alle Strategien von P_1 bewahrenden Austauschbarkeitsbegriff gegen einen Service P_2 austauscht, ist sichergestellt, dass jeder Service, der P_1 verwenden konnte, auch P_2 verwenden kann. Damit bietet das Unternehmen seinen Kunden die Zuverlässigkeit, den vom Unternehmen angebotenen Service auch nach einem Austausch verwenden zu können.

Je mehr Strategien allerding von P_1 bewahrt bleiben sollen, gegen desto weniger Services darf das Unternehmen seinen Service P_1 austauschen. Das bedeutet umgekehrt, je weniger Strategien von P_1 bewahrt bleiben müssen, desto mehr Freiheiten gewinnt das Unternehmen, P_1 zu verändern.

Es ist anzunehmen, dass sich ein Unternehmen möglichst viel Freiheit beim Austausch seines Service P_1 wünscht. Da Unternehmen aber oft mehr als einen Kunden haben, wird es nicht reichen wie in [Ric02] durch den Austausch von P_1 gegen P_2 nur eine bestimmte Strategie von P_1 zu bewahren. Stattdessen möchte ein Unternehmen vermutlich vor allem eine ausgewählte Menge $\mathcal{R}$ von Strategien von P_1 bewahren, in den meisten Fällen wahrscheinlich seine Kunden, die P_1 tatsächlich verwenden. Auf diese Weise gibt das Unternehmen seinen Kunden, die P_1 tatsächlich verwenden, die Sicherheit, auch den neuen Service P_2 verwenden zu können, während das Unternehmen gleichzeitig zusätzliche Freiheiten in der Wahl von P_2 gewinnt.

Genau auf dieses Interesse zielt der Austauschbarkeitsbegriff ab, den wir in diesem Abschnitt definieren und untersuchen werden. Dieser parametrisierte Austauschbarkeitsbegriff ist bezüglich einer Menge $\mathcal{R}$ von Serviceautomaten formuliert. Wenn P_1 der urprüngliche Serviceautomat und P_2 der geänderte ist, dann sei $\mathcal{R}$ eine Menge von Strategien von P_1. Dann nennen wir P_1 bezüglich $\mathcal{R}$ gegen P_2 austauschbar, wenn jede Strategie aus $\mathcal{R}$ auch Strategie von P_2 ist.

Definition 20 (austauschbar bezüglich $\mathcal{R}$)
Seien P_1 und P_2 zwei Serviceautomaten, sei $\mathcal{R} = \{R_1, \ldots, R_n\}$ eine endliche Menge k-beschränkter Strategien von P_1. Dann heißt P_1 *bezüglich $\mathcal{R}$ gegen P_2 k-beschränkt austauschbar*, wenn für alle $R \in \mathcal{R}$ gilt: $R \in \mathit{Strat}_k(P_2)$.

Weiterhin ist $\mathit{Exchange}_k(P_1, \mathcal{R}) = \{\, P_2 \mid \forall R \in \mathcal{R} : R \in \mathit{Strat}_k(P_2) \,\}$ *die Menge aller Serviceautomaten, gegen die P_1 bezüglich $\mathcal{R}$ k-beschränkt ausgetauscht werden kann.* ⌟

Für ein Unternehmen, das seinen Service P_1 bezüglich einer Menge von Kunden $\mathcal{R}$ gegen einen Service P_2 austauschen will, ist also die Menge $\mathit{Exchange}_k(P_1, \mathcal{R})$ interessant. Ist P_2 in $\mathit{Exchange}_k(P_1, \mathcal{R})$, dann kann das Unternehmen seinen Service P_1 bedenkenlos gegen P_2 austauschen, denn es kann sicher sein, dass die Kunden in $\mathcal{R}$ weiterhin P_2 verwenden werden können.

Um zu überprüfen, ob P_2 in $\mathit{Exchange}_k(P_1, \mathcal{R})$ enthalten ist, kann das Unternehmen gemäß der Definition von $\mathit{Exchange}_k(P_1, \mathcal{R})$ vorgehen: Es kann für jeden einzelnen Serviceautomaten R aus $\mathcal{R}$ prüfen, ob er Strategie von P_2 ist. Sind sie es alle, dann gehört P_2 zu $\mathit{Exchange}_k(P_1, \mathcal{R})$ und das Unternehmen kann P_1 gegen P_2 austauschen.

Falls das Unternehmen tatsächlich nur herausfinden möchte, ob es einen Service P_1 durch *einen* bestimmten Service P_2 austauschen kann, ist dieses Vorgehen durchaus sinnvoll.

Die Situation ändert sich allerdings, wenn das Unternehmen viele Services P_2 als Ersatz für P_1 untersuchen möchte. Dann ist es sehr aufwendig, für jeden dieser vielen Services zu überprüfen, ob er eine Strategie aller Services aus $\mathcal{R}$ ist. Wünschenswert wäre, *einmal* eine Darstellung berechnen zu können, die die gemeinsamen Strategien aller Services aus $\mathcal{R}$ beschreibt. Dann müsste man jeden Austauschkandidaten P_2 nur *einmal* mit dieser Darstellung vergleichen, um zu entscheiden, ob er eine Strategie *aller* Services aus $\mathcal{R}$ ist und damit P_1 ersetzen kann.

Wünschenswert wäre auch, wenn man bei der Wahl von P_2 nicht auf probieren angewiesen wäre, wenn es also eine Möglichkeit gäbe, ein P_2 gezielt so zu konstruieren, dass es zu $\mathit{Exchange}_k(P_1, \mathcal{R})$ gehört.

Beide Wünsche lassen sich mit dem Konzept der Bedienungsanleitung und ihrem im vorigen Kapitel definierten Produkt erfüllen. Wir müssen nur eine Bedienungsanleitung finden, die alle Serviceautomaten in $\mathit{Exchange}_k(P_1, \mathcal{R})$ charakterisiert. Anhand derer könnten wir dann leicht entscheiden, ob ein gegebener Serviceautomat in $\mathit{Exchange}_k(P_1, \mathcal{R})$ enthalten ist oder gezielt einen solchen konstruieren.

Die Menge $\mathit{Exchange}_k(P_1, \mathcal{R})$ umfasst alle Serviceautomaten, die Strategien aller Serviceautomaten in $\mathcal{R}$ sind. Deshalb bilden wir das Produkt der Bedienungsanleitungen aller Serviceautomaten aus $\mathcal{R}$, um eine die Menge $\mathit{Exchange}_k(P_1, \mathcal{R})$ charakterisierende Bedienungsanleitung zu erhalten. Jeder Serviceautomat P_2, der dieser Bedienungsanleitung genügt, kann dann als Ersatz für den ursprünglichen Serviceautomaten P_1 von allen Serviceautomaten aus $\mathcal{R}$ verwendet werden. Dieser Zusammenhang ist in Satz 3 formuliert.

Satz 3
Seien P_1 und $\mathcal{R}$ wie in Defintion 20. Sei $\mathcal{S} = \{S_1^{\Phi_1}, \ldots, S_n^{\Phi_n}\}$ eine endliche Menge

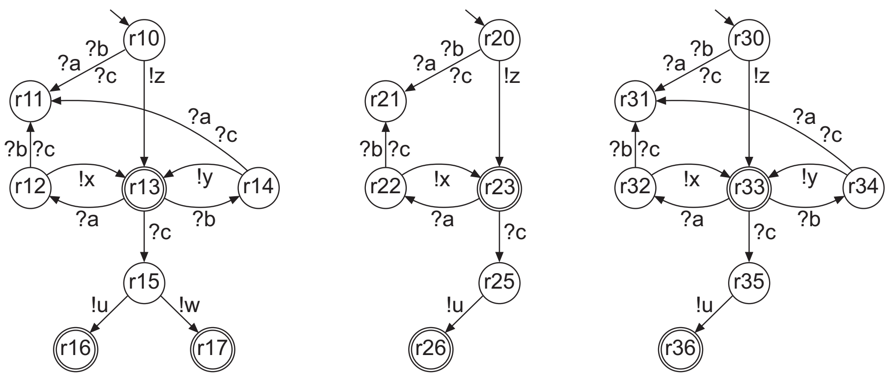
von Bedienungsanleitungen, wobei jedes $S_i^{\Phi_i}$ eine k-beschränkte Bedienungsanleitung für R_i ist. Sei $\mathcal{S}_\otimes^\Phi$ das Produkt von $\mathcal{S}$. Dann gilt $\mathit{Exchange}_k(P_1, \mathcal{R}) = \mathit{Comply}(\mathcal{S}_\otimes^\Phi)$. ⌟

Beweis 5
Seien P_1, $\mathcal{R}$, $\mathcal{S}$ und $\mathcal{S}_\otimes^\Phi$ wie im Satz. Dann gilt

$$\begin{aligned}
\mathit{Exchange}_k(P_1, \mathcal{R}) &= \{\, P_2 \mid \forall R \in \mathcal{R} : R \in \mathit{Strat}_k(P_2) \,\} \\
&= \{\, P_2 \mid \forall R \in \mathcal{R} : P_2 \in \mathit{Strat}_k(R) \,\} \\
&= \bigcap_{R \in \mathcal{R}} \mathit{Strat}_k(R) \\
&= \mathit{Comply}(S_\otimes^\Phi) \qquad \text{(wegen Lemma 5).}
\end{aligned}$$

□

Nehmen wir zum Beispiel an, ein Unternehmen bietet den Service P_1 in Abbildung 3.4 aus dem vorigen Kapitel an. Er ähnelt dem Serviceautomaten P aus Abbildung 2.1, den wir bereits in Kapitel 2 kennengelernt haben, ist aber noch einfacher. Das Unternehmen besitze zudem drei Kunden R_1, R_2 und R_3, die in Abbildung 4.2 dargestellt sind. R_1 und R_2 kennen wir bereits aus der Abbildung 3.1 in Kapitel 3. Der Serviceautomat R_3 ist eine Abwandlung dieser beiden. Alle drei Serviceautomaten R_1, R_2 und R_3 sind Strategien von P_1. Das Unternehmen möchte seinen Service P_1 gegen einen anderen Service austauschen, und zwar so, dass die Services R_1, R_2 und R_3 seiner Kunden auch Strategien des neuen Services sind.



(a) R_1 (b) R_2 (c) R_3

Abbildung 4.2: Verschiedene Serviceautomaten.

Um die Bedienungsanleitung zu erhalten, die alle Serviceautomaten charakterisiert, gegen die das Unternehmen den Serviceautomaten P_1 bezüglich $\mathcal{R} = \{R_1, R_2, R_3\}$ austauschen kann, konstruieren wir zuerst die Bedienungsanleitungen OG_{R_1}, OG_{R_2} und OG_{R_3} von R_1, R_2 bzw. R_3 und anschließend ihr Produkt.

Da wir in diesem Beispiel nur an 1-beschränkten Strategien interessiert sind, bilden wir OG_{R_1}, OG_{R_2} und OG_{R_3} für $k = 1$. OG_{R_1} und OG_{R_2} kennen wir bereits aus Kapitel 3, wo sie in Abbildung 3.2 in der gewohnten vereinfachten Weise dargestellt sind. OG_{R_3} ist ihnen ähnlich und in Abbildung 4.3 zu finden.

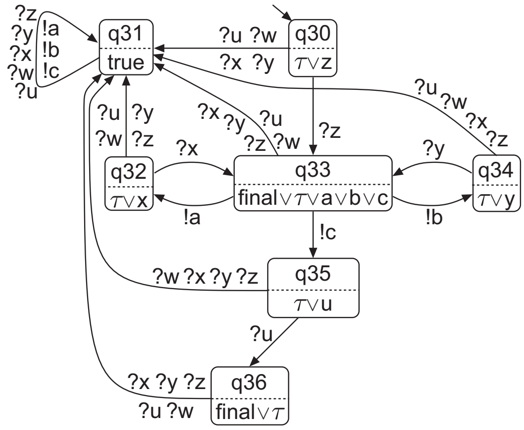


Abbildung 4.3: Die Bedienungsanleitung für R_3 für $k = 1$.

Die Produktbedienungsanleitung $OG_{\mathcal{R}} = OG_{R_1} \otimes OG_{R_2} \otimes OG_{R_3}$ ist in Abbildung 4.4 dargestellt. Da sie aus den 1-beschränkten Bedienungsanleitungen OG_{R_1}, OG_{R_2} und OG_{R_3} entstanden ist, charakterisiert sie alle Serviceautomaten, die 1-beschränkte Strategien von R_1, R_2 und R_3 sind, und damit genau diejenigen Serviceautomaten, gegen die man P_1 bezüglich $\mathcal{R}$ 1-beschränkt austauschen kann ($Exchange_1(P_1, \mathcal{R})$).

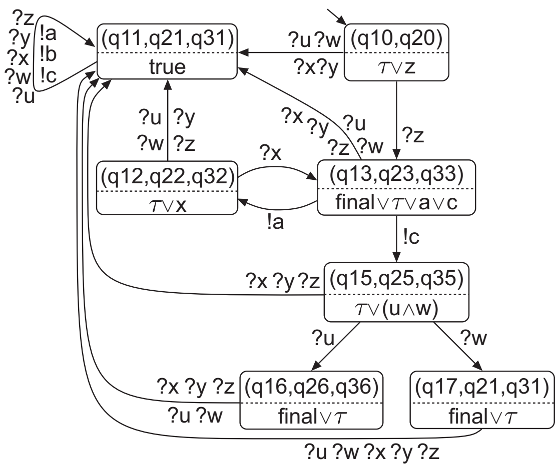


Abbildung 4.4: Die Bedienungsanleitung $OG_{R1} \otimes OG_{R2} \otimes OG_{R3}$ für $k = 1$.

Zum Beispiel genügt der Serviceautomat P aus Abbildung 2.1, den wir schon aus Kapitel 2 kennen, der Bedienungsanleitung $OG_{\mathcal{R}}$. Das heißt, P ist eine 1-

beschränkte Strategie von R_1, R_2 und R_3 und damit für P_1 bezüglich $\mathcal{R}$ 1-beschränkt austauschbar. Auch intuitiv ist klar, dass man den Serviceautomaten P_1 eine im Zustand p13 beginnende und aus dem Senden von a und dem Empfangen von x bestehende Schleife, die der einzige Unterschied zwischen P und P_1 ist, hinzufügen darf und R_1, R_2 und R_3 dabei Strategien von P_1 bleiben. Schließlich enthalten R_1, R_2 und R_3 jeweils die komplementäre Schleife.

Im Gegensatz dazu genügt der Serviceautomat P_2 aus Abbildung 3.5 *nicht* der Bedienungsanleitung $OG_{\mathcal{R}}$. Er genügte auch schon nicht der Bedienungsanleitung OG_{R_2} aus Abbildung 3.2(b), war deshalb keine Strategie von R_2 aus Abbildung 4.2(b) und ist deshalb nicht für P_1 austauschbar. Dies liegt daran, dass P_2 aus dem Zustand p23 heraus ein b senden kann, was R_2 nicht bereit ist, zu empfangen.

Das Unternehmen könnte sich deshalb schnell dafür entscheiden, die Strategie R_2 von P doch nicht bewahren zu wollen, und P nur bezüglich der Menge $\mathcal{R}' = \{R_1, R_3\}$ austauschen zu wollen. Dann bildet es die Produktbedienungsanleitung $OG_{R_1} \otimes OG_{R_3}$ (siehe Abbildung 4.5), und stellt fest, dass dann $P_2 \in \mathit{Comply}(OG_{R_1} \otimes OG_{R_3})$ und in dieser Situation auch P_2 für P austauschen könnte.

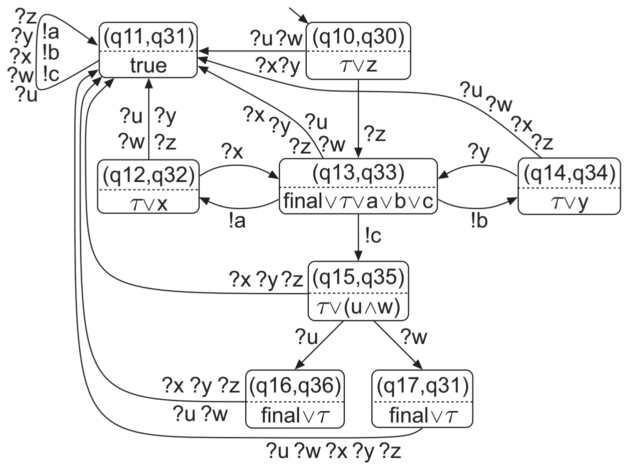


Abbildung 4.5: Die Bedienungsanleitung $OG_{R1} \otimes OG_{R3}$ für $k = 1$.

Darüber hinaus könnte das Unternehmen auch gezielt einen Serviceautomaten konstruieren, der gegen P bezüglich $\mathcal{R}'$ austauschbar ist. Dafür müsste es nur mit einem zu $OG_{R_1} \otimes OG_{R_3}$ isomorphen Serviceautomaten starten und aus ihm Kanten und Zustände höchstens so entfernen, dass alle Zustandsbeschriftungen von $OG_{R_1} \otimes OG_{R_3}$ erfüllt bleiben. So lange das der Fall ist, würde jeder so konstruierte Serviceautomat der Bedienungsanleitung $OG_{R_1} \otimes OG_{R_3}$ genügen, also Strategie von R_1 und R_3 und damit für P bezüglich $\mathcal{R}'$ austauschbar bleiben.

5 Bedienungsanleitungen aus Teilautomaten konstruieren

Nachdem wir sie im vorigen Kapitel benutzt haben, um austauschende Serviceautomaten zu charakterisieren, kommen wir nun zu einer ganz anderen Anwendung von Produktbedienungsanleitungen. Es geht darum, die Bedienungsanleitung eines gegebenen Serviceautomaten P zu konstruieren, indem wir P in Teilautomaten zerlegen und dann das Produkt der Bedienungsanleitungen dieser Teilautomaten bilden. Dieser Ansatz könnte helfen, effizienter als auf die in Kapitel 2 definierte allgemeine Art die Bedienungsanleitung eines großen, aber stark modularisierten Service nach einer Änderung an einem seiner Module zu konstruieren.

Stellen wir uns einen großen Service P vor, der, weil er systematisch entworfen wurde, aus vielen wohlstrukturiert verbundenen Modulen besteht. Seine Bedienungsanleitung nach der in Kapitel 2 beschriebenen Art zu konstruieren kann mit wachsender Größe des Service und seiner Schnittstelle schnell sehr aufwendig werden. Wir erinnern uns, dass dazu erst die kanonische Strategie von P konstruiert werden muss, deren Zustände aus Situationsmengen bestehen, die beschreiben, in welchen Zuständen sich P und die Nachrichtenkanäle nach einer bestimmten Ereignissequenz befinden können. Beginnend vom Anfangszustand aus muss wiederholt der Effekt aller möglichen Sende- und Empfangsereignisse auf die neu erreichten Situationsmengen berechnet werden. Jeder Zustand q der so entstehenden kanonischen Strategie von P wird anschließend mit einer booleschen Formel beschriftet, die sich aus der Situationsmenge $K(q)$ berechnet. Die so beschriftete kanonische Strategie von P ist dann die Bedienungsanleitung von P. Da nur die Topologie und die Beschriftung der Bedienungsanleitung interessant sind, werden die berechneten Situationsmengen üblicherweise am Ende der Konstruktion verworfen.

Nach einer Änderung an P müssen bisher alle eben beschriebenen Berechnungen erneut ausgeführt werden, um die Bedienungsanleitung von P zu konstruieren. Um diese unnötige Neuberechnung zu verringern, schlagen wir einen alternativen Ansatz zur Konstruktion der Bedienungsanleitung von P vor. Dabei zerlegen wir den Serviceautomaten P geeignet in Teilautomaten und konstruieren anschließend das Produkt derer Bedienungsanleitungen, um eine Bedienungsanleitung von P zu erhalten. Die Bedienungsanleitungen der Teilautomaten konstruieren wir auf herkömmliche Weise oder durch weitere Zerlegung und Produktbildung.

Wenn nun ein Teilautomat von P geändert wird, brauchen wir nur noch die Bedienungsanleitung des geänderten Teilautomaten zu konstruieren und anschließend das Produkt mit den noch unverändert vorhandenen Bedienungsanleitungen der restlichen Teilautomaten zu bilden, um die Bedienungsanleitung von P zu erhalten.

Da auf diese Weise viele Situationsmengen nicht neu berechnet werden müssen und die Produktbildung von Bedienungsanleitungen nur vergleichsweise einfache syntaktische Operationen verwendet, versprechen wir uns, durch diesen Ansatz die Bedienungsanleitung von P nach einer Änderung an P schneller berechnen zu können.

Wir werden diesen Ansatz zuerst an einem Beispiel demonstrieren, bevor wir ihn anschließend formalisieren.

Betrachten wir den Serviceautomaten P aus Abbildung 5.1. Von seinem Anfangszustand `p0` aus kann er einen internen Schritt nach `p1` oder `p2` ausführen. Je nach dem für welchen Schritt er sich nichtdeterministisch entscheidet, sendet er danach entweder ein `a` und erwartet anschließend ein `x`, oder sendet ein `b` und erwartet ein `y`. Nachdem er das `x` bzw. `y` empfangen hat, geht er durch einen internen Schritt in seinen Endzustand `p7` über.

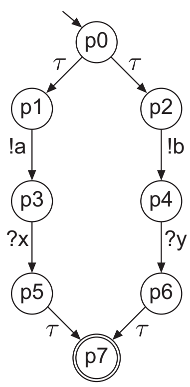


Abbildung 5.1: Ein Serviceautomat P.

Unser Ziel ist, eine Bedienungsanleitung für P zu konstruieren, um alle Strategien von P zu charakterisieren.

Wie wir bereits erläutert haben, entscheidet sich P in seinem Anfangszustand nichtdeterministisch und ohne Einfluss der Umgebung, ob er in den linken im Zustand `p1` beginnenden Zweig oder den rechten in `p2` beginnenden Zweig wechselt. Am Ende führen beide Zweige in den einzigen Endzustand `p7` von P. Das bedeutet, dass jede Strategie von P mit beiden möglichen Entscheidungen von P umgehen können muss.

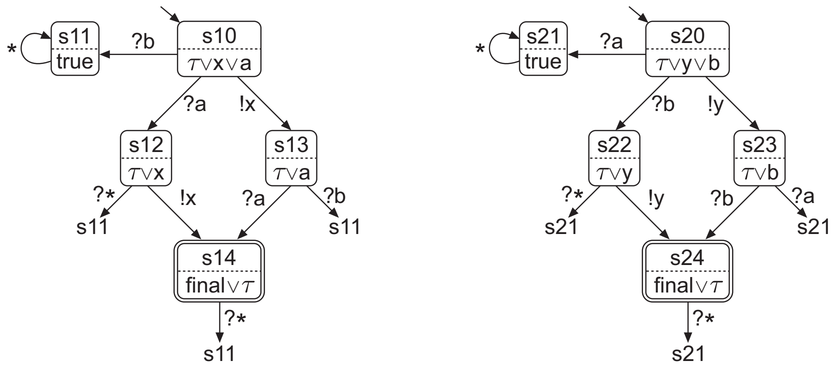
Wenn wir nun beide Zweige von P geeignet als zwei separate Serviceautomaten P_1 und P_2 auffassen, dann ist jeder Serviceautomat genau dann Strategie von P, wenn er Strategie von P_1 und Strategie von P_2 ist. Die Strategien von P_1 werden durch seine Bedienungsanleitung OG_{P_1}, die Strategien von P_2 durch OG_{P_2} charakterisiert. Das Produkt $OG_{P_1} \otimes OG_{P_2}$ dieser beiden Bedienungsanleitungen charakterisiert genau diejenigen Serviceautomaten, die Strategie von P_1 und P_2 und damit auch Strategie von P sind. Also ist $OG_{P_1} \otimes OG_{P_2}$ eine geeignete Bedienungsanleitung von P.

In unserem Beispiel eignet sich die Zerlegung von P in die in Abbildung 5.2 dargestellten Serviceautomaten P_1 und P_2. Dabei werden die ersten und letzten Zustände des linken und rechten Zweigs von P zu den Anfangs- bzw. Endzuständen von P_1 bzw. P_2.

(a) P_1 (b) P_2

Abbildung 5.2: P zerlegt in zwei Serviceautomaten P_1 und P_2.

Die sich ergebenden Bedienungsanleitungen OG_{P_1} und OG_{P_2} von P_1 bzw. P_2 sind in Abbildung 5.3 dargestellt. Wieder wurden für die Darstellung τ-Schleifen, Livelocks und überflüssige Literale in den booleschen Formeln weggelassen.



(a) OG_{P_1} (b) OG_{P_2}

Abbildung 5.3: Die Bedienungsanleitungen für P_1 und P_2.

Abbildung 5.4 zeigt die Produktbedienungsanleitung $OG_{P_1} \otimes OG_{P_2}$. Sie charakterisiert genau die Serviceautomaten, die Strategie von P_1 und Strategie von P_2 sind. Aus ihrem Anfangszustand $(\mathsf{s10}, \mathsf{s20})$ mit der Beschriftung $\tau \vee (\mathsf{a} \wedge \mathsf{b})$ lesen wir ab, dass jede Strategie von P_1 und P_2 in ihrem Anfangszustand nach beliebig vielen τ-Schritten sowohl ein a, als auch ein b empfangen können muss. Weiterhin lesen wir ab, dass sie nach dem Empfang von a ein x und nach dem Empfang von b ein y senden und dadurch jeweils in einen Endzustand übergehen muss. Dieses Verhalten muss auch jede Strategie von P zeigen. Und tatsächlich ist genau jeder durch $OG_{P_1} \otimes OG_{P_2}$ charakterisierte Serviceautomat eine Strategie von P, $OG_{P_1} \otimes OG_{P_2}$ also eine Bedienungsanleitung von P.

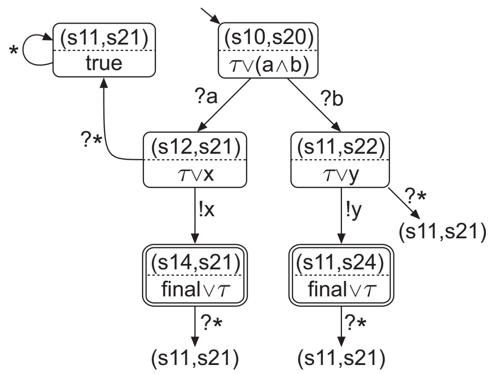


Abbildung 5.4: Die Produktbedienungsanleitung $OG_{P_1} \otimes OG_{P_2}$.

An einem weiteren Beispiel sehen wir, dass dieses Prinzip auch funktioniert, wenn der gegebene Serviceautomat, zu dem wir eine Bedienungsanleitung konstruieren wollen, keine (livelockfreie) Strategie besitzt.

Abbildung 5.5 zeigt einen Serviceautomaten R, der sich vom eben betrachteten Serviceautomaten P aus Abbildung 5.1 nur durch die Reihenfolge der Sende- und Empfangstransitionen unterscheidet. Nach seiner nichtdeterministischen Entscheidung, ob er von seinem Anfangszustand r0 aus in den Zustand r1 oder r2 wechselt, wartet er erst auf die Nachricht x oder y und sendet nach deren Empfang ein a bzw. b.

Offensichtlich hat R keine livelockfreie k-beschränkte Strategie, da ein Partner von R nicht wissen kann, für welchen Zweig sich R entscheidet. Damit kann ein Partner von R nicht wissen, ob er ihm ein x oder ein y senden soll. Er muss allerdings eines von beiden senden, damit R überhaupt in seinen Endzustand gelangen kann. Beide Nachrichten zu senden wäre aber auch falsch, weil eine von beiden auf jeden Fall im Nachrichtenkanal liegen bliebe.

Wenn wir nun R nach demselben Prinzip wie P zerlegen, erhalten wir die in Abbildung 5.6 dargestellten Serviceautomaten R_1 und R_2.

Die Bedienungsanleitungen OG_{R_1} und OG_{R_2} von R_1 bzw. R_2 sind in Abbildung 5.7 dargestellt.

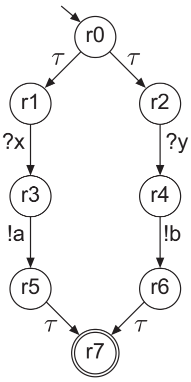
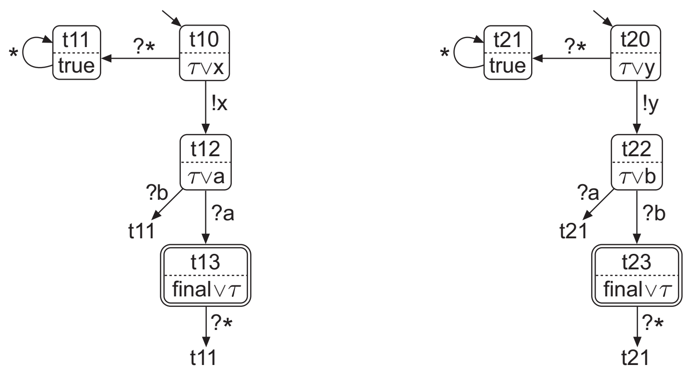


Abbildung 5.5: Ein Serviceautomat R.

(a) R_1

(b) R_2

Abbildung 5.6: R zerlegt in zwei Serviceautomaten R_1 und R_2.



(a) OG_{R_1}

(b) OG_{R_2}

Abbildung 5.7: Die Bedienungsanleitungen für R_1 und R_2.

Aus ihnen errechnet sich die in Abbildung 5.8 dargestellte Produktbedienungsanleitung $OG_{R_1} \otimes OG_{R_2}$. Da nur den Anfangszustand t10 von OG_{R_1} eine !x-Kante verlässt, nicht aber den Anfangszustand t20 von OG_{R_2}, verlässt auch keine !x-Kante den Anfangszustand (t10, t20) von $OG_{R_1} \otimes OG_{R_2}$. Aus analogem Grund besitzt der Zustand (t10, t20) auch keine ausgehende !y-Kante. Da die Beschriftung $\tau \vee (\mathsf{x} \wedge \mathsf{y})$ von (t10, t20) aber (abgesehen vom Livelocks erlaubenden τ) die Anwesenheit einer !x- und einer !y-Kante fordert, kann kein livelockfreier Serviceautomat die Beschriftung von (t10, t20) erfüllen. Also können wir aus der Bedienungsanleitung $OG_{R_1} \otimes OG_{R_2}$ ablesen, dass R_1 und R_2 keine gemeinsame livelockfreie Strategie besitzen und damit auch R keine livelockfreie Strategie hat. Das heißt, auch hier war unser Ansatz, eine Bedienungsanleitung von R aus seinen Teilautomaten zu konstruieren erfolgreich.

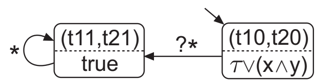


Abbildung 5.8: Die Produktbedienungsanleitung $OG_{R_1} \otimes OG_{R_2}$.

Alle gezeigten Beispiele folgen demselben Muster: Der betrachtete Serviceautomat begibt sich von seinem Anfangszustand aus nichtdeterministisch durch einen internen Schritt in einen seiner voneinander unabhängigen Zweige, die alle in seinen einzigen Endzustand münden. Wie man beim Vorliegen dieses Musters den betrachteten Serviceautomaten in Teilautomaten zerlegt und anschließend durch Produktbildung seine Bedienungsanleitung konstruiert, formalisieren wir in Regel 1.

Regel 1
Seien im Folgenden $P = (Q, I, O, \delta, q_0, F)$, $P_1 = (Q_1, I_1, O_1, \delta_1, q_{01}, F_1)$ und $P_2 = (Q_2, I_2, O_2, \delta_2, q_{02}, F_2)$ drei schnittstellengleiche Serviceautomaten, für die gilt:

- $Q_i \subseteq Q$ für $i = 1, 2$
- $Q_1 \cap Q_2 = \emptyset$
- $Q_i \neq \emptyset$ für $i = 1, 2$
- $\delta_i = \delta \cap Q_i \times (I \cup O \cup \{\tau\}) \times Q_i$ für $i = 1, 2$
- $|F| = |F_1| = |F_2| = 1$; sei $F = \{q_F\}$, $F_1 = \{q_{F_1}\}$ und $F_2 = \{q_{F_2}\}$
- $Q \setminus (\{q_0\} \cup F \cup Q_i) = Q_j$ für $i, j = 1, 2$ und $i \neq j$
- $\delta \cap (\{q_0\} \times (I \cup O \cup \{\tau\}) \times Q) = \{(q_0, \tau, q_{01}), (q_0, \tau, q_{02})\}$
- $\delta \cap (Q \times (I \cup O \cup \{\tau\}) \times F) = \{(q_{F_1}, \tau, q_F), (q_{F_2}, \tau, q_F)\}$
- für alle $q, q' \in Q$, $x \in I \cup O \cup \{\tau\}$ gilt:
  - $(q, x, q') \in \delta$ impliziert $q' \neq q_0$
  - Wenn $(q, x, q') \in \delta$ und $q \in Q_i$ ist, dann ist $q' \notin Q_j$ für $i, j = 1, 2$ und $i \neq j$.

Seien OG_{P_1} und OG_{P_2} die k-beschränkten Bedienungsanleitungen von P_1 bzw. P_2. Dann gilt: $\mathit{Comply}(OG_{P_1} \otimes OG_{P_2}) = \mathit{Strat}_k(P)$. ⌟

Wir lassen Regel 1 hier ohne Beweis stehen und begnügen uns damit, sie an den obigen Beispielen und durch ihre strengen Voraussetzungen plausibel gemacht zu haben. Regel 1 ist außerdem so angelegt, dass sie leicht auf n viele Zweige im zu zerlegenden Serviceautomaten P verallgemeinert werden kann.

Wegen der strengen Voraussetzungen von Regel 1 ist es auf jeden Fall nötig weitere Regeln zusammenzutragen, um den hier dargestellten Ansatz, eine Bedienungsanleitung aus Teilautomaten zu konstruieren, bei realistischen Services intensiv anwenden zu können. Schließlich können Module in Services auf verschiedene Weise miteinander verknüpft sein.

Zusätzlich könnte man nach Transformationsregeln für Serviceautomaten suchen, die beschreiben, wie man einen Serviceautomaten schrittweise in einen anderen umformen kann, ohne die Bedienungsanleitung des Serviceautomaten zu verändern. Auf diese Weise könnten eventuell die für den vorgeschlagenen Ansatz nötigen Muster in Serviceautomaten hergestellt werden.

6 Fazit

In der vorliegenden Arbeit haben wir ein Produkt auf Bedienungsanleitungen definiert, seine Eigenschaften untersucht und zur Lösung verschiedener Fragestellungen über Services verwendet. Dabei haben wir das Produkt zweier Bedienungsanleitungen OG_1 und OG_2 bewusst so definiert, dass es den Durchschnitt der von OG_1 und OG_2 beschriebenen Serviceautomaten charakterisiert. Damit verhält es sich analog zum Produkt traditioneller endlicher Automaten. Denn das Produkt zweier endlicher Automaten A_1 und A_2 ist ein endlicher Automat, dessen Sprache genau der Durchschnitt der Sprachen von A_1 und A_2 ist.

Die Hauptanwendung von Produktbedienungsanleitungen betraf die Frage, gegen welche Services P' sich ein gegebener Service P bezüglich einer gegebenen Menge $\mathcal{R}$ von Strategien von P austauschen lässt, so dass P' Strategie aller Services aus $\mathcal{R}$ ist. Die Menge aller möglichen solchen P' konnten wir durch das Produkt der Bedienungsanleitungen aller Services aus $\mathcal{R}$ charakterisieren.

Außerdem haben wir einen Ansatz vorgeschlagen, das Produkt von Bedienungsanleitungen zu verwenden, um die neue Bedienungsanleitung eines großen, stark modularisierten Serviceautomaten P nach einer Änderung an P eventuell effizienter als herkömmlich zu konstruieren. Dafür zerlegen wir P in Teilautomaten und bilden das Produkt derer Bedienungsanleitungen, um eine Bedienungsanleitung für P zu erhalten.

Die Praxistauglichkeit beider Anwendungen bleibt zu zeigen. Mit `fiona` [LMSW06] existiert ein Werkzeug, das Bedienungsanleitungen für Serviceautomaten berechnen kann. Es kann außerdem überprüfen kann, ob ein gegebener Serviceautomat einer gegebenen Bedienungsanleitung genügt. Die Bildung von Produktbedienungsanleitungen in `fiona` zu implementieren ist damit ohne großen Aufwand möglich.

Danach könnten Produktbedienungsanleitungen prinzipiell sofort benutzt werden, um im Rahmen der durch das Konzept der Bedienungsanleitung vorgegebenen Möglichkeiten, die in dieser Arbeit untersuchte Frage der Austauschbarkeit eines Service P bezüglich einer Menge von Strategien von P für echte Services zu beantworten. Dass diese Frage für Unternehmen interessant sein könnte, haben wir in Kapitel 4 erörtert. Ob dem tatsächlich so sein und ob sich die hier vorgeschlagene Lösung als praxistauglich erweisen wird, bleibt sowohl abzuwarten, als auch Quell weiterer Arbeit.

Um allerdings den Ansatz zur Konstruktion von Bedienungsanleitungen aus Teilautomaten praktisch nutzen zu können, sind weitere theoretiche Vorarbeiten nötig. In dieser Arbeit haben wir nur *eine* Regel präsentiert, die ein bestimmtes Muster, nach dem Module von Services verknüpft sein können, behandelt. Regeln für weitere Muster zu finden, ist die Voraussetzung dafür, anschließend die Praxistauglichkeit des Ansatzes an realistischen Beispielen überprüfen zu können.

Außerdem erscheint es lohnend weitere Analogien zwischen Bedienungsanleitungen und traditionellen endlichen Automaten zu erforschen. Wenn Operationen auf Bedienungsanleitungen zur Verfügung stünden, mit denen sich zusätzlich zum Durchschnitt, die Vereinigung und das Komplement der durch die beteiligten Bedienungsanleitungen charakterisierten Serviceautomaten beschreiben ließen, könnten weitere Fragestellungen bezüglich Services eventuell elegant durch geschickte Verknüpfung ihrer Bedienungsanleitungen gelöst werden.

Literaturverzeichnis

[Aal98] AALST, W. M. P. van der: The Application of Petri Nets to Workflow Management. In: *The Journal of Circuits, Systems and Computers* 8 (1998), Nr. 1, S. 21–66

[Aal99] AALST, W. M. P. van der: Interorganizational Workflows: An Approach based on Message Sequence Charts and Petri Nets. In: *Systems Analysis - Modelling - Simulation* 34 (1999), Nr. 3, S. 335–367

[AB02] AALST, W. M. P. van der ; BASTEN, T.: Inheritance of workflows: an approach to tackling problems related to change. In: *Theoretical Computer Science* 270 (2002), Nr. 1–2, S. 125–203

[BCPV04] BROGI, Antonio ; CANAL, Carlos ; PIMENTEL, Ernesto ; VALLECILLO, Antonio: Formalizing Web Service Choreographies. In: *Electr. Notes Theor. Comput. Sci.* 105 (2004), S. 73–94

[Fer04] FERRARA, Andrea: Web services: a process algebra approach. In: AIELLO, Marco (Hrsg.) ; AOYAMA, Mikio (Hrsg.) ; CURBERA, Francisco (Hrsg.) ; PAPAZOGLOU, Mike P. (Hrsg.): *ICSOC*, ACM, 2004. – ISBN 1–58113–871–7, S. 242–251

[Gla01] VAN GLABBEEK, Rob J.: The Linear Time – Branching Time Spectrum I; The Semantics of Concrete, Sequential Processes. In: BERGSTRA, J.A. (Hrsg.) ; PONSE, A. (Hrsg.) ; SMOLKA, S.A. (Hrsg.): *Handbook of Process Algebra*. Elsevier, 2001. – Available at `http://boole.stanford.edu/pub/spectrum1.ps.gz`, Kapitel 1, S. 3–99

[Got00] GOTTSCHALK, Karl: Web Services Architecture Overview. / IBM developerWorks. 2000. – IBM Whitepaper. `http://ibm.com/developerWorks/web/library/w-ovr/`

[HDA+05] HIDDERS, Jan ; DUMAS, Marlon ; AALST, Wil M. P. van der ; TER HOFSTEDE, Arthur H. M. ; VERELST, Jan: When are two workflows the same? In: *CRPIT '41: Proceedings of the 2005 Australian symposium on Theory of computing*. Darlinghurst, Australia, Australia : Australian Computer Society, Inc., 2005. – ISBN 1–920682–23–6, S. 3–11

[HU79] HOPCROFT, John E. ; ULLMAN, Jeffrey D.: *Introduction to automata theory, languages, and computation*. Addison-Wesley, 1979. – ISBN 0–201–02988–X

[KHA03] Kiepuszewski, Bartek ; ter Hofstede, Arthur H. M. ; Aalst, Wil M. P. van der: Fundamentals of control flow in workflows. In: *Acta Informatica* 39 (2003), Nr. 3, S. 143–209

[LMSW06] Lohmann, Niels ; Massuthe, Peter ; Stahl, Christian ; Weinberg, Daniela: Analyzing Interacting BPEL Processes. In: *Business Process Management, 4th International Conference, BPM 2006, Vienna, Austria, September 5-7, 2006, Proceedings* Bd. 4102, Springer-Verlag, September 2006, S. 17–32

[LMW06] Lohmann, Niels ; Massuthe, Peter ; Wolf, Karsten: Operating Guidelines for Finite-State Services / Humboldt-Universität zu Berlin. 2006 (210). – Informatik-Berichte

[Mar03] Martens, Axel: *Verteilte Geschäftsprozesse - Modellierung und Verifikation mit Hilfe von Web Services*, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät II, Dissertation, 2003. – erschienen in WiKi: Stuttgart, Berlin & Paris

[Mil71] Milner, Robin: An Algebraic Definition of Simulation Between Programs. In: *IJCAI*, 1971, S. 481–489

[Mil89] Milner, R.: *Communication and concurrency*. Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 1989. – ISBN 0–13–115007–3

[MRS05] Massuthe, Peter ; Reisig, Wolfgang ; Schmidt, Karsten: An Operating Guideline Approach to the SOA. In: *Annals of Mathematics, Computing & Teleinformatics* 1 (2005), Nr. 3, S. 35–43

[Par81] Park, David: Concurrency and Automata on Infinite Sequences. In: *Proceedings of the 5th GI-Conference on Theoretical Computer Science*. London, UK : Springer-Verlag, 1981. – ISBN 3–540–10576–X, S. 167–183

[Rei86] Reisig, Wolfgang: *Petrinetze: Eine Einführung*. 2. Auflage. Springer-Verlag, Januar 1986. – ISBN 3–540–16622–X

[Ric02] Richter, Wolf: *Spezifikation und Implementation organisationsübergreifender Geschäftsprozesse mit Petrinetzen*, Humboldt-Universität zu Berlin, Diplomarbeit, 2002

[RKM04] Rao, Jinghai ; Küngas, Peep ; Matskin, Mihhail: Logic-based Web Services Composition: From Service Description to Process Model. In: *ICWS '04: Proceedings of the IEEE International Conference on Web Services (ICWS'04)*. Washington, DC, USA : IEEE Computer Society, 2004. – ISBN 0–7695–2167–3, S. 446

Erklärung

Hiermit erkläre ich, die vorliegende Arbeit „Produktbedienungsanleitungen zur Charakterisierung austauschbarer Services“ selbstständig und ohne fremde Hilfe verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet zu haben.

Weiterhin erkläre ich hiermit mein Einverständnis, dass die vorliegende Arbeit in der Bibliothek des Instituts fur Informatik der Humboldt-Universität zu Berlin ausgestellt werden darf.

Berlin, den 13. März 2007