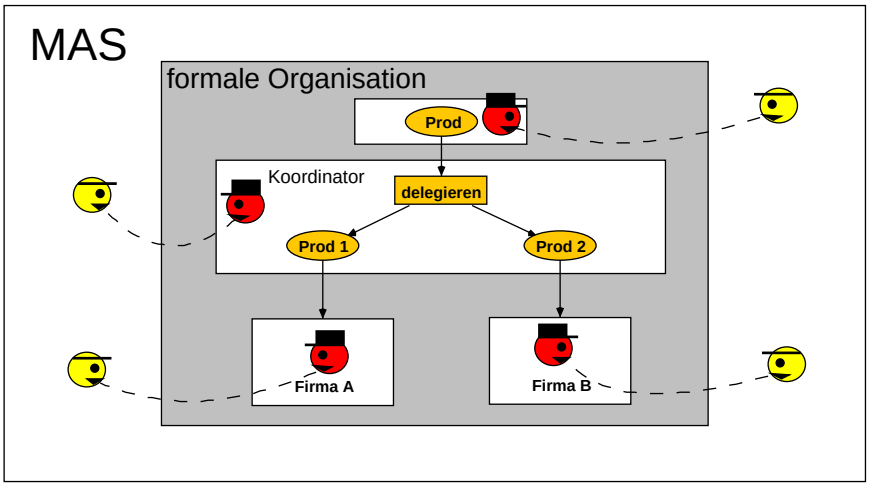
Prüfungsunterlagen zur Vorlesung »Modellierung und Petrinetze« im Wintersemester 2007/2008

Eine formale Spezifikation reflexiver Selbstorganisation in Multiagentensystemen

Michael Köhler

Stand vom 24. Januar 2008



1 Koordinierung in Agentensystemen

In diesem Kapitel betrachten wir die grundlegenden Konzepte von Multiagentensystemen. Wir geben zunächst eine Einführung in die bestehenden Ansätze und betrachten dann, wie sich die derzeitige Forschung auf den Gegenstand der Koordination bezieht. Wir betrachten dazu Kooperations- und Verhandlungsprotokolle, normativ handelnde Agenten, Organisationsmodelle von Agentensystemen und die Formen der sozialen Selbstorganisation.

1.1 Grundelemente der Multiagentensysteme

In diesem Kapitel betrachten wir die Sichtweise der *verteilten künstlichen Intelligenz* (VKI) (engl. distributed artificial intelligence, DAI) auf Multiagentensysteme. Nach Weiß (1999) widmet sich die VKI der Analyse, der Konstruktion und der Anwendendung von Multiagentensystemen, d.h. von Systemen, in denen mehrere interagierende, intelligente Agenten eine Menge von Zielen verfolgen oder Aufgaben ausführen.[1]

In dieser Definition sind bereits die zentralen Aspekte von Agentensystemen enthalten: Erstens handelt es sich bei den Systemkomponenten um Agenten, d.h. um lose gekoppelte Einheiten, die im Rahmen ihrer Möglichkeiten autonom handeln. Zweitens sind die Systemeinheiten intelligent, worunter man grob verstehen kann, dass Agenten in der Lage sind, auch in verschiedenen Umgebungen flexibel zu agieren, dass sie in der Lage sind, rational zu planen, und dass sie die Fähigkeit besitzen zu lernen. Drittens spielt die Interaktion der Agenten eine zentrale Rolle, da nicht nur die Einheiten verteilt sind, sondern auch Fähigkeiten, Daten und Ressourcen.

Hier zeigt sich die konzeptionelle Abgrenzung der VKI zur KI. Die künstliche Intelligenz (KI) stellt mit dem Bereich des intelligente Problemlösens eines einzelnen situierten Systems die kognitive Aspekte in den Vordergrund. Für die VKI besteht der Kontext eines Agenten nun wiederum aus Agenten. Damit steht hier der Interaktionszusammenhang zwischen den Agenten im Vordergrund. Intelligenz des Ganzen ergibt sich aus dem Zusammenspiel der Teile. Dieser Paradigmenwechsel geht einher mit einer Neuausrichtung in Hinblick auf die Nachbardisziplinen. Während die KI der Psychologie und der Neurologie nahesteht, orientiert sich die VKI in Richtung der Sozial- und Wirtschaftswissenschaften.

Die Fokussierung auf die Interaktionszusammenhänge und ihre Strukturen hat den interessanten Aspekt, dass sie zweiseitig ausgerichtet ist. Zum einen hat sie eine Ausrichtung auf die Mikro-Ebene, d.h. den Agenten, der in die Interaktionszusammenhänge eingebettet ist und dessen Handeln somit kontexttualisiert wird. Genauso werden aber mit ihr auch die Makro-Zusammenhänge auf Ebene des Gesamtsystems betrachtet, bei denen nicht mehr die Handlungen des einzelnen relevant sind, sondern vielmehr der Prozess mit seinen Mustern insgesamt. Diese Mikro-Makro-Dualität wird uns in den weiteren Ausführungen begleiten.

[1] "DAI is the study, construction, and application of multiagent systems, that is, systems in which several interacting, intelligent agents pursue some set of goals or perform some tasks." (Weiß, 1999, S.1)

Die Möglichkeit – und meist auch der Zwang – zur Interaktivität der Agenten bildet das Leitmotiv für die VKI-Forschung: „Wer kommuniziert wann mit wem worüber?“ Wie im folgenden noch weiter ausgeführt wird, kann man die Koordination der Agenten grob in zwei Klassen aufteilen: in Kooperation und in Konkurrenz. Ersteres liegt vor, wenn die Agenten gemeinsam auf ein Ziel hin arbeiten und sich nur abstimmen müssen, letzteres liegt vor, wenn die Ziele in Konflikt miteinander stehen.

Die Koordination wird durch Protokolle, d.h. durch verteilte Algorithmen geregelt. Typischerweise ist aus der Protokollsicht jeder Agent des Systems ein potentieller Interaktionspartner. Offensichtlich steigt die Komplexität, geeignete Interaktionspartner zu finden, die Ziele mit ihnen abzustimmen und die Pläne konfliktfrei zu gestalten, mit der Anzahl der in einem System vorhandenen Agenten an. Ziel muss es aber sein, den Aufwand linear, d.h. in $O(n)$ zu gestalten, da man dann die Reaktionszeiten des Systems auch bei steigenden Agentenzahlen konstant halten kann, indem man die Zahl der Rechner im MAS-Netzwerk entsprechend aufstockt. Der Aufwand wächst im allgemeinen jedoch schneller als die Zahl der Agenten selbst, da typische verteilte Algorithmen Kommunikationskomplexitäten von $O(n \log n)$ oder $O(n^2)$ aufweisen. Dieser Zuwachs an Koordinierungsaufwand ist als *Skalierungsproblematik* der VKI bekannt. Ihr kann nur begegnet werden, indem man Strukturen schafft, die den Koordinationsprozess regeln, beispielsweise, indem man nur eine Teilmenge aller Agenten betrachtet oder nur bestimmte Teams für Aufgaben heranzieht.

Die Skalierungsproblematik wirft somit Fragen auf, die über die Ebene einzelner Agenten hinausgeht und sich nur auf der Systemebene behandeln lässt. Es ist daher wichtig, sich mit der Formation von Teams und der Organisationsstruktur von Multiagentensystemen zu beschäftigen.

Einige Autoren gehen sogar noch weiter, indem sie postulieren, dass die Skalierungsproblematik nur dann zu lösen ist, wenn die Organisationsstruktur explizit repräsentierter Bestandteil des Systems ist und das Systems aktiv auf diese Einfluss nehmen kann.

> Thus, the current state of the art is challenged by MASs that are larger or where the magnitude or speed of agent population variability confounds one overall design. To tackle both these problems we hypothesize that MASs should be *self-building* (able to determine the most appropriate organizational structure for the system by themselves at run-time) and *adaptive* (able to change this structure as their environment changes). (Turner und Jennings, 2001, 1)

Diese Eigenschaft der Selbstkonstruktion lässt sich auch als *Reflexivität* des Multiagentensystems beschreiben, bei dem die Organisationsstruktur das veränderte Objekt ist. Dies ist ein echtes Novum, denn in den meisten Forschungsbeiträgen erscheinen Organisationsstrukturen als ein kontextuelles Konstrukt, innerhalb dessen sich die Agenten koordinierten. Der Agent war also die betrachtete Variable, die sich in Abhängigkeit von der Organisation entwickelte: Der Agent war in Teams involviert, welche von der Organisationsstruktur geformt werden; er richtet sein Handeln an den Notwendigkeiten aus, die sich aus der Organisations- und Teamstruktur ergeben usw. Bei all diesen Formen variiert also der Agent, und die Organisation bleibt statisch. Reflexivität kehrt jetzt diese Perspektive um, indem sie die Organisation zur Variablen erklärt, die durch das Handeln der Akteure geformt wird. Man

geht sogar so weit, in diesem Zusammenhang von der *lernenden Organisation* zu sprechen.

Diese zutiefst sozialwissenschaftlich orientierte Sichtweise, dass Agenten soziale Akteure sind, bestimmt die folgende Darstellung insofern, als dass sie sich an dem Aspekt der *Sozialität* des Agenten in den verschiedenen Ausprägungen orientiert. Sozialität findet ihren Ausdruck auf den verschiedensten Betrachtungsebenen eines Agentensystems. Wir betrachten im folgenden den Agenten in seiner Rolle als sozialen *Akteur*, die Dynamik, die *Agentengruppen* entfalten, die Rolle von *Organisationsformen*, die eine vorstrukturierende Funktion für Agentengruppen besitzen und schließlich die Ebene der *Agentensozialität*, die dadurch gekennzeichnet ist, dass die Organisationsformen nicht statisch fixiert sind, sondern sich aus dem Zusammenspiel der Akteure ergeben.

1.1.1 Agenten als soziale Akteure

Agenten sind in einer Umgebung situierte Einheiten, die ihre Aufgaben autonom verfolgen. Ein Agent ist autonom, d.h. im Rahmen der operationalen Randbedingungen (interne wie externe) ist er in seiner Handlungswahl frei, Restriktionen sind nur von ihm selbst auferlegte. Ein Agent ist in dem Sinne intelligent, als dass er zielverfolgend ist und dass hiermit eine gewisse Flexibilität bezüglich der Zielrealisierung verbunden ist, so dass ein Agent auch auf sich verändernde Randbedingungen reagieren kann. Insofern unterscheidet sich ein Agent der VKI noch nicht von einem Agenten der KI. Hier trifft man nicht die Annahme, dass die Umgebung aus anderen Agenten besteht. Aus der Tatsache, dass ein Agent in der Lage ist, auf die Umwelt einzuwirken, folgt dann, dass er in der Lage ist, mit anderen Agenten zu interagieren.

Diese Eigenschaften grenzen Agenten von Objekten ab, die weder autonom noch lernend oder zielverfolgend sind. Die engste Verbidung finden Agenten noch zu den aktiven Objekten (Nierstrasz, 1993), bei denen jedem Objekt ein eigener Kontrollfluss (engl. thread) zugeordnet wird. Aktive Objekte kann man eine gewisse Zielverfolgung zusprechen. Objekte sind aber nicht autonom, denn sie diskriminieren im allgemeinen nicht danach, von welchem Objekt ihre Methoden aufgerufen werden. Ein Agent kann dies tun, beispielsweise um Konflikte zu vermeiden. Ebenso unterscheiden sich Agenten von Expertensystemen, da letztere nicht sozial eingebettet sind.

Es gibt unterschiedliche Forschungsansätze, den Aufbau eines Agenten zu konzeptionieren. Da Zielverfolgung und Planung ein zentraler Gegenstand der KI ist, entstammen viele Strukturierungsansätze aus dem Bereich der formalen Logik. Hier werden Modallogiken zur Wissensrepräsentation (siehe Levesque und Lakemeyer, 2000) oder zur Spezifikation der temporalen Aktivitäten (siehe Wooldridge und Jennings, 1995; Wooldridge, 2000) verwendet. Als praktische Systeme, die auf diesem Ansatz aufbauen, sind hier METAMEM (Barringer u. a., 1990) und Congolog (Giacomo u. a., 2000) zu nennen.

Ein weit verbreiteter Ansatz ist das Design von Agenten nach dem BDI-Paradigma (Bratman, 1987; Rao und Georgeff, 1991), bei dem das Wissen (engl. beliefs), die Wünsche (engl. desires) sowie die Absichten (engl. Intentions) eines Agenten spezifiziert werden und die dann zur Wahl der Aktivitäten herangezogen werden. Die verbreiteste Datenstruktur zur Darstellung von Zielen und Plänen basiert auf der formalen Logik, hier speziell der Modallogik. Modalitäten sind Operatoren der Logik, die sich auf den Wahrheitswert von Formeln beziehen. Dies ist

in der Prädikatenlogik nicht möglich, da man in ihr keine Prädikate über Formeln ausdrücken kann. Der folgende Ausruck ist demnach ungeeignet, um auszudrücken, dass Alice glaubt, dass Tweety ein Vogel sei:

$$\text{believes}(\textit{alice}, \text{is-a-bird}(\textit{tweety}))$$

Das Problem besteht darin, dass die Argumente des Prädikats *believes* Terme sein müssen, is-a-bird(*tweety*) jedoch eine Formel ist.

Die klassischen Modalitäten sind die der Notwendigkeit und die der Möglichkeit (Hughes und Cresswell, 1984). Eine Formel ϕ ist notwendigerweise wahr, wenn man sich keine Welt vorstellen kann, in der ϕ nicht wahr wäre. Diese Tatsache wird durch die Formel $\Box\phi$ notiert. Die Aussage ϕ ist nur möglicherweise wahr, wenn Welten vorstellbar sind, in denen ϕ wahr ist, aber auch welche, in denen das nicht gilt. Dies wird als $\Diamond\phi$ notiert. Erweitert man die Aussagenlogik um diese beiden Modalitäten, so erhält man die propositionale Modallogik. Um die Semantik der Modalitäten exakt festzuhalten, fordert man axiomatisch die Gültigkeit gewisser Formeln (vgl. Tabelle 1.1). Die Axiome K und N werden stets gefordert, die übrigen nicht notwendigerweise. Verwendet man Kripke-Strukturen als Modell der Modallogik, so korrespondieren die Axiome mit Eigenschaften der zu betrachtenden Sichtbarkeitsrelation der Kripke-Struktur. Sie ist in der dritten Spalte notiert.

Name	Axiom	Bedingung
K	$\Box(\phi \Longrightarrow \psi) \Longrightarrow (\Box\phi \Longrightarrow \Box\psi)$	
N	$\phi \Longrightarrow \Box\phi$	
T	$\Box\phi \Longrightarrow \phi$	reflexiv
D	$\Box\phi \Longrightarrow \Diamond\phi$	seriell
4	$\Box\phi \Longrightarrow \Box\Box\phi$	transitiv
5	$\Diamond\phi \Longrightarrow \Box\Diamond\phi$	Euklid

Tabelle 1.1: Axiome der Modallogik

Notwendigkeit ist nicht die einzige mögliche Interpretation der Modalität $\Box$. Interpretiert man die Modalität $\Box\phi$ als „ϕ wird gewusst“, so kann man hiermit eine Logik des Glaubens/Wissens formalisieren. Dies hat zur Konsequenz, dass aus dem Notwendigkeitsaxioms $\phi \Longrightarrow \Box\phi$ folgt, dass *alle* Tautologien gewusst werden. Außerdem ist Wissen unter Implikation abgeschlossen, d.h. alle gültige Folgerungen $\phi_1 \wedge \ldots \wedge \phi_n \Longrightarrow \phi$ werden gewusst. Gilt das D-Axiom ($\Box\phi \Longrightarrow \Diamond\phi$ und $\Diamond\phi \equiv \neg\Box\neg\phi$), so ist Wissen widerspruchsfrei. Gilt das T-Axiom ($\Box\phi \Longrightarrow \phi$), so können nur Wahrheiten gewusst werden. Gilt das 4-Axiom ($\Box\phi \Longrightarrow \Box\Box\phi$), so besteht Wissen über das Gewusste. Gilt das 5-Axiom ($\neg\Box\neg\phi \Longrightarrow \Box(\neg\Box\neg\phi)$), so besteht Wissen über das Ungewusste.

In der *BDI-Logik* werden die Überzeugungen (*beliefs*), die Bedürfnisse (engl. *desires*) und die Absichten eines Agenten (engl. *intentions*) eines Agenten durch Modalitäten modelliert. Wir folgen der Darstellung von Panzarasa u. a. (2002). Überzeugungen betreffen die Welt, die mentalen Haltungen anderer Agenten oder auch den Agenten selbst. Die Tatsache, dass der Agent a zum Zeitpunkt t der Überzeugung ist, dass ϕ gilt, wird als $\mathbf{Bel}(a, \phi)(t)$ dargestellt. Es gelten die Axiome KD45:

$$\begin{aligned}
\mathbf{Bel}(a,\phi)(t) \wedge \mathbf{Bel}(a,(\phi \Longrightarrow \psi))(t) &\Longrightarrow \mathbf{Bel}(a,\psi)(t)\\
\mathbf{Bel}(a,\phi)(t) &\Longrightarrow \neg\mathbf{Bel}(a,\neg\phi)(t)\\
\mathbf{Bel}(a,\phi)(t) &\Longrightarrow \mathbf{Bel}(a,\mathbf{Bel}(a,\phi))(t)\\
\neg\mathbf{Bel}(a,\phi)(t) &\Longrightarrow \mathbf{Bel}(a,\neg\mathbf{Bel}(a,\phi))(t)
\end{aligned}$$

Die Bedürfnisse und Ziele eines Agenten drücken die Zustände der Welt aus, die er gern erreichen möchte. Die Tatsache, dass der Agent a hat zum Zeitpunkt t das Bedürfnis hat, ϕ zu erreichen, wird als $\mathbf{Des}(a, \phi)(t)$ dargestellt, Ziele als $\mathbf{Goal}(a, \phi)(t)$. Der Unterschied zwischen Zielen und Bedürfnisse liegen darin, dass Bedürfnisse inkonsistent und sogar unerfüllbar sein können, während Ziele als Menge *konsistenter und realisierbarer* Zustände definiert sind, von denen man annehmen kann, dass der Agent sie erreichen kann. Agenten haben keine den Überzeugungen widersprechenden Ziele.

Die Absichten eines Agenten sind immer auf einen bestimmten erstrebenswerten Zustandes der Welt ausgerichtet. Besitzt ein Agent die individuelle Absicht, einen Zustand zu erreichen, so ist er sich selbst gegenüber verpflichtet (*self-commitment*), entsprechend zu handeln. Die Absicht des Agenten a, zum Zeitpunkt t einen Zustand, in dem ϕ gilt, anzustreben, wird $\mathbf{Int}(a, \phi)(t)$ notiert. Ein Agent besitzen keine Absichten, die seinen Überzeugungen widersprechen:

$$\mathbf{Int}(a, \phi)(t) \Longrightarrow \neg\mathbf{Bel}(a, \neg\phi)(t)$$

Eine deutliche Abkehr von planenden Ansätzen stellt das reaktive Paradigma dar, bei dem Agenten kein Weltmodell aufbauen, keine – oder nur geringe – Zustandsinformation speichern und ihre Wahl anhand einer Liste einfacher Wenn-Dann Regeln treffen. Die Vorteile des Ansatzes sind seine geringe Darstellungsgröße und seine kurzen Reaktionszeiten. Anwendung findet dieser Ansatz daher insbesondere im Bereich der Robotik (Brooks, 1990). Besondere Prominenz hat dieser Ansatz unter dem Schlagwort der Schwarmintelligenz bekommen (Kennedy und Eberhart, 2001). Ausgangspunkt ist hier die Feststellung, dass Tierpopulationen (wie Ameisenkolonien, Fisch- oder Vogelschwärme) in der Lage sind, global kohärentes (um nicht zu sagen: intelligentes) Verhalten aufweisen, und dies obwohl das einzelne Individuum dazu kognitiv nicht in der Lage ist. So kann eine einzelne Ameise keine Wegewahloptimierung bei der Futtersuche betreiben, die Kolonie als ganzes schon.

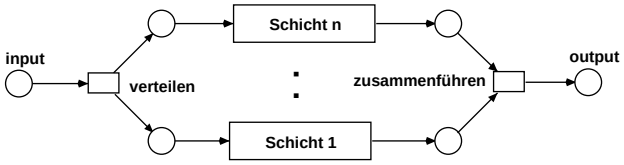


Abbildung 1.1: Horizontale Agentenarchitektur

Weitere typische Ansätze stellen die Schichtenarchitekturen dar. Hierbei unterscheidet man zwischen sogenannten horizontalen und vertikalen Schichtungen. Bei der horizontalen Schichtung erhalten alle Planungsschichten die gleiche Eingabe. Ihre Ergebnisse werden dann zu einer Gesamtentscheidung zusammengesetzt. Ein Beispiel für eine horizontale Schichtung ist das Touring-Machines-System (Ferguson, 1992).

Vertikale Architekturen bearbeiten die Eingabe und geben ihre Ergebnisse an die nächste Ebene weiter. Man unterscheidet Systeme, bei denen die Bearbeitung in der letzten Schicht zu Ende ist (one-pass), von solchen, bei denen die letzte Schicht ihre Ergebnisse wieder zurück reicht, so dass sie die gesamte Schichtung in rückwärtiger Richtung durchläuft (two-pass). Ein Beispiel für eine vertikale Schichtung ist das INTERARP-System (Müller und Pischel, 1994).

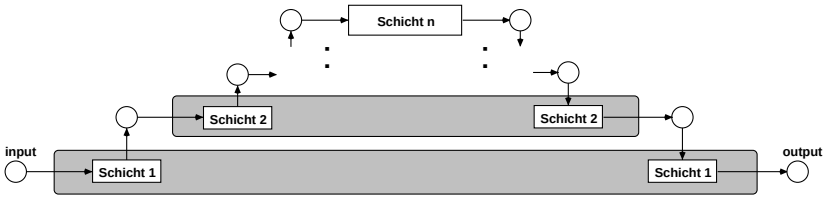


Abbildung 1.2: Vertikale Agentenarchitektur (two-pass)

Daneben existieren als Erweiterung der objektorientierten Programmiersprachen auch agentenorientierte Sprachen wie Agent-0 (Shoham, 1990), Agentspeak (Rao, 1996), JASON (Bordini u. a., 2007) oder JADE (Bellifemine u. a., 2001).

1.1.2 Interaktionsgruppen

Agenten existieren nicht isoliert, sondern stets im Kontext anderer Agenten. Agenten interagieren, daher ist die Perspektive des Interaktionszusammenhangs der Agenten, kurz: seine *Interaktionsgruppe*, eine natürliche Betrachtungsebene des sozialen Agenten. Durch den Aspekt der *Sozialität* von Agenten geht das Agentenkonzept der VKI über das der KI hinaus. So spricht sich Gasser (1991) dafür aus, die Konzepte des Wissens und der Handlung speziell vor dem Hintergrund ihrer sozialen Bezüge zu interpretieren. Dies legt nahe, das Charakteristische von Agenten (wie *Autonomie*, *Flexibilität*, *Intelligenz* usw.) nicht primär als eine interne Eigenschaften zu deuten, sondern es vielmehr als ein relationales Konstrukt zu betrachten, das sich z.B. in den Interaktionsbeziehungen als Abhängigkeitsverhältnis äußert. Abhängigkeitsverhältnisse ergeben sich hierbei beispielsweise aus der funktionalen Spezialisierung der Agenten oder dem Besitz relevanter Ressourcen.

Interaktionsgruppen ergeben sich durch die sozialen Verflechtungen ihrer Agenten. Diese können funktionaler Natur sein, d.h. ein Agent ist auf die Fähigkeiten eines anderen angewiesen, um seine eigenen Aufgaben zu erfüllen, oder koordinativer Natur, beispielsweise, wenn es um die Vermeidung von Zugriffskonflikten geht.

Die Interaktion von Agenten wird in Multiagentensystemen meist in Form von Nachrichtenkommunikation modelliert. Um eine Kommunikation von Sender und Empfänger zu ermöglichen ist es notwendig, dass sich beide Parteien über die Bedeutung der Nachricht einigen. Dazu ist erstens eine gemeinsame Darstellungssprache notwendig, zweitens eine Taxonomie (auch: Ontologie), die das Diskursuniversum beschreibt, sowie drittens eine Beschreibung der Intention, die mit der Nachricht verbunden ist. Eine Spezifikationssprache für diese drei Elemente wird als *Agentenkommunikationssprache* bezeichnet.

Ontologien oder Konzeptbeschreibungssprachen finden sich im Kontext des *semantic web* (Studer u. a., 2003). Beispiele für Ontologiesprachen sind DAML (Ogbuji und Ouellet, 2002) oder OWL (Smith u. a., 2004). Als formales Konzept der Kommunikation greift die VKI auf die Theorie der Sprechakte (engl. speech act theory) nach Searle (1970) zurück, nach der Kommunikationen Handlungen sind, die den Empfänger zu einer bestimmten Reaktion veranlassen sollen. Sprechakte sind informierend, anfragend, fordernd usw. Beispiele für Agentenkommunikationssprache sind die Spezifikationssprachen KQML (*knowledge query and manipulation language*) (Knowledge Sharing Initiative External Interfaces Working Group, 1993) und FIPA-ACL (*FIPA agent communication language*) (FIPA, 1998).

Neben der Spezifikation der Kommunikation ist noch der darauf aufbauende Ko-

ordinationsprozess zu charakterisieren. Man kann dabei grob zwei Klassen unterscheiden: In die erste Klasse fallen solche Ansätze, die sich eines Mediums bedienen, um die Agenten zu koordinieren. Als klassischen Ansatz sind hier die *blackboard*-Systeme zu nennen, bei denen Agenten Nachrichten wie an einem Schwarzen Brett hinterlassen. Ein Agent, der eine Teilaufgabe delegieren möchte, schreibt die Aufgabenbeschreibung an das Schwarze Brett. Alle anderen Agenten können dann diese Nachricht sehen und bei Interesse die erarbeitete Lösung wieder an Brett hängen. Eine direkte Kommunikation ist daher nicht notwendig. Ein Beispiel für eine solche Architektur sind die *tuple spaces* in LINDA (Carriero und Gelernter, 1989). In die zweite Klasse fallen solche Ansätze, bei denen Ablaufprotokolle die direkte Kommunikation der Agenten regeln. Hier wird spezifiziert, wer wem Nachrichten schicken darf, oder welche Informationen die jeweiligen Nachrichten enthalten müssen. Klassische Beispiele sind hier das *Contract-Net* Protokoll (Smith, 1977) und das *Partial Global Planning* Protokoll (Durfee und Lesser, 1991).

1.1.3 Organisationsformen

In der allgemeinen Formulierung ergeben sich die Kommunikationsbeziehung anhand der möglichen Delegationsbeziehungen und anhand der funktionalen oder temporalen Abhängigkeiten. Alle Agenten des Systems sind dabei zu berücksichtigen, denn im schlimmsten Fall hängen alle Agenten von einander ab.

In natürlichen Sozialsystemen treffen die Akteure jedoch nicht beliebig aufeinander, und sie müssen auch nicht alle ihre Pläne mit allen anderen bis ins kleinste abstimmen. Dies liegt daran, dass soziale Systeme bereits vorstrukturiert sind. Es existieren Strukturen, die regeln, welche Interaktionsgruppen sich bilden können. Daneben existieren Normen in Form von *Rollen*, die unabhängig von ihrem Träger Rechte und Verpflichtungen beschreiben. Damit reduziert sich der Abstimmungsaufwand der Einzelpläne enorm.

Die Einschränkungen, die soziale Strukturen für die Agenten darstellen, stehen nicht im Widerspruch zur Autonomie, denn obwohl die Agenten den Erwartungen einer Rolle genügen müssen, so sind sie doch prinzipiell in ihrer Wahl, ob sie eine Rolle einnehmen, frei. Natürlich sind die Agenten nur so frei, wie es die Abhängigkeiten zulassen.

Beispiele für Sozialstrukturen sind Organisationstrukturen, die durch Hierarchien und Kompetenzen die Menge der potentiellen Teammitglieder eingrenzen und gleichzeitig deren Verhalten bereits grob festlegen. Die Abläufe, die die Zusammenarbeit von Rolleninhabern regeln, sind meist durch Geschäftsprozesse (engl. „business processes“) standardisiert.[2]

Diese sozialen Strukturen, die unabhängig von den Akteuren existieren, bilden die Grundlage für die Koordinationsfähigkeit eines Multiagentensystems. Man kann dabei grob in subjektive und objektive Formen der Koordination unterscheiden (Schumacher, 2001). Die subjektive Form stellt dabei die Perspektive der Agenten auf den koordinationsprozess dar, die objektive Form die Perspektive die des gesamten Systems.

Als zentraler Schlüssel zur Koordinierung wird aus Agentensicht das Konzept der *Zusicherung* (engl. commitment) gesehen. Durch Zusicherungen legen Agenten ihr zukünftiges Verhalten gegenüber anderen fest, üblicherweise unter der Bedingung, dass gewisse Rahmenbedingungen in der Zwischenzeit unverändert bleiben. Darauf

[2] In der Literatur werden solche Systeme auch unter dem Begriff der elektronischen Institution oder unter dem des elektronischen Marktes behandelt.

aufbauend gelangt man von individuellen Annahmen, Wünschen und Absichten zu den innerhalb einer Gruppen geteilten Annahmen, Wünschen und Absichten. Die Gruppenmitglieder legen einander insbesondere durch die gemeinsamen Absichten fest, indem sie ihre Absprachen in einem überindividuellen Element, den geteilten Absichten (engl. joint intentions), bündeln (vgl. Jennings, 1993, 1996; Castelfranchi, 1995; Ossowski, 1999).

Die Systemsicht betrachtet dagegen die Interaktionsstruktur an sich, losgelöst von der Agentenperspektive. Die konkreten Realisierungformen der Organisationsstrukturen werden im Rahmen der *Organisationstheorie* untersucht, ein Forschungsbereich, der sowohl im Fokus der Betriebs- als auch der Sozialwissenschaften steht und in jüngster Zeit auch durch den Bereich der Wirtschaftsinformatik thematisiert wird. Eine Wissenschaftsdisziplin im Schnittpunkt der drei Forschungsbereiche ist die *Computational Organsisation Theory* (siehe Prietula u. a., 1998; Carley und Gasser, 1999).

1.1.4 Sozialität als Mikro/Makro Wechselwirkung

Es ist zu beachten, dass Organisationen keine festgefügten Strukturen sind, innerhalb deren sich die Tätigkeiten der Agenten abspielen und von der alle Agenten restriktiv gelenkt werden.[3] Organisationen treten den Akteuren zwar einerseits als überindividuelle, quasi-objektivierte Strukturen mit eigener Realität entgegen, sie sind umgekehrt aber auch sozial konstituiert, indem die Akteure die Strukturen durch ihre Interaktionen erst (re-)produzieren. Es wird auch davon gesprochen, dass die Systemstrukturen ein emergentes Produkt der Handlungen des Systems selbst sind. Dies hat man sich so vorzustellen, dass Agenten durch ihre Handlungen innerhalb der Organisation ständig die Ausrichtung oder die Gewichtung der Organisationsziele verschieben, meist ohne dies intendiert zu haben. Auch Rollenerwartungen oder Prozeduren können sich so verschieben.

Man kann also nicht davon ausgehen, dass Organisationsstrukturen ein fester Kontext wäre, der unter anderem für effektive Kommunikation sorgt (wie in Shoham und Tennenholtz, 1994). Man kann aber auch nicht umgekehrt davon ausgehen, dass soziale Strukturen emergent aus dem Nichts entstehen (wie in Walker und Wooldridge, 1995). Es ist vielmehr so, dass beide Variablen – Organisationsstruktur und Akteurshandeln – wechselseitig miteinander in Verbindung stehen. Darüber sollte zudem nicht vergessen werden, dass Organisationen ihrerseits einen gesellschaftlichen Kontext besitzen, also ihrerseits eingebettet sind.

Diese Wechselseitigkeit ist als die Mikro-Makro-Dualität bekannt. Die adäquate Modellierung der Wechselwirkung beider Elemente gilt als Schlüssel zur Konstruktion skalierungsfähiger Multiagentensystem (vgl. Schillo u. a., 2000).

Die *wechselseitige* Beeinflussung interaktionistischer und strukturalistischer Elemente als Prozess dynamisiert das System enorm. Dies stellt eine Herausforderung für das Design von Agentensystemen dar, denn das Wechselspiel von Struktur und Handlung setzt eine dynamische Anpassbarkeit von Systemstrukturen und -verhalten voraus. Diese Strukturdynamik wird in der Informatik meist vermieden, da derartig selbstmodifizierende Systeme erfahrungsgemäß schwerer zu beherrschen sind als gesteuerte oder starre Systeme. Die Anforderungen heutiger Anwendungskontexte erfordern jedoch in zunehmendem Maße die Selbstmodifikation, da auch die Anwendungskontexte starken Veränderungen unterworfen sind. Damit stellt sich erstens die Frage nach den zugrundeliegenden Mechanismen der Strukturdynamik

[3] Soziologisch formuliert: Organisationen sind nicht nur strukturalistisch geprägte Phänomene.

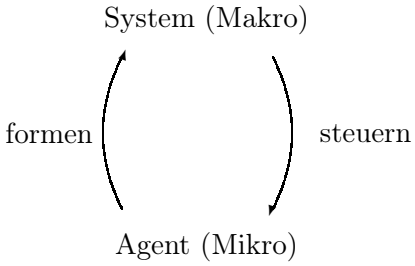


Abbildung 1.3: Mikro/Makro Wechselwirkung

und zweitens, wie eine zielgerichtete Steuerung solch flexibler Systeme erreicht werden kann.

1.1.5 Multiagentensysteme versus Workflows

Die Metaphern des Multiagentensystems sind eng mit denen der betrieblichen Informationssysteme (engl. workflow management system, WFMS) verwandt. Die Agentenmetapher beschreibt Systeme als System von Agenten, die aufgrund ihrer Einschränkungen (Wissen, Ressourcen, Fähigkeiten) mit anderen Agenten interagieren müssen, wobei die Agenten einerseits um Ressourcen in Konkurrenz zueinander stehen und andererseits in Team kooperieren müssen, um ihre Ziele verfolgen zu können. Die Agentenmetapher stellt damit einen bottom-up Ansatz dar. Betriebliche Informationssysteme dagegen nehmen eine top-down Perspektive ein: Workflow-Management-Systeme bestehen aus Datenbanken und -schemata zusammen mit einem Rollenmodell zur Zugriffkontrolle und zur Definition von Verantwortlichkeiten. Darauf aufbauend werden die Geschäftsprozesse definiert, die vom System zur Laufzeit überwacht werden.

Vergleicht man Multiagentensysteme mit Workflow-Management-Systemen, so stellt man fest, dass mit beiden Ansätzen jeweils Vor- und Nachteile verbunden sind. Die Agentenmetapher stellt ein gutes Konzept dar, um die Entwurfsprinzipien der Abstraktion und der Dekomposition zu unterstützen. Stellt man sich Agenten als Spezialisten vor, so werden für einen Bereich Wissen und Fähigkeiten an jeweils einem Ort gebündelt, so dass sich Vorteile für die Wartbarkeit des Systems ergeben. Außerdem erhöht es die Flexibilität des Systems, denn um neue Fähigkeiten bereitzustellen, ist es ausreichend, einen Agenten zu implementieren, der die Funktionalität dann anderen Agenten zur Verfügung stellt. Sind die Fähigkeiten formal spezifiziert, kann das Systems die neue Funktionalität sogar automatisch integrieren. Die Agentenmetapher ist jedoch ungeeignet, um die Systemstrukturierung anzuleiten. Hier sind überindividuelle Konstrukte – wie Teams, Allianzen usw. – notwendig. Geht man von sich autonom koordinierenden Agenten aus, so erzeugt das Koordinationsverhalten der Agenten emergentes Gesamtverhalten. Prominentestes Beispiel ist der Bereich der Schwarmintelligenz. Im allgemeinen ist dies aus Entwicklersicht jedoch unzulänglich, denn wenn zur Designzeit unbekannt ist, welche Koordinationsmuster entstehen, dann ist auch nicht klar, ob diese erwünschtes (genauer: korrektes) Verhalten darstellen.

Bei den Workflow-Mangement-Systeme stellt sich die Situation annähernd komplementär dar. Das System hat die volle Kontrolle über die Daten und die Geschäftsprozesse, wodurch es überhaupt erst möglich wird, gewünschte Systemeigenschaften zuzusichern, darunter so essentielle Sicherheitsaspekte wie Integrität, Ver-

traulichkeit, Verläßlichkeit. Die Zusicherbarkeit wird jedoch meist dadurch erkauft, dass diese Systeme keine oder nur geringe Mechanismen zur flexiblen Anpassung des Systems bieten. Anpassungen sind Wartungstätigkeiten der Entwicklung. Auch sind diese Systeme meist geschlossene Systeme, die keine Unterstützung organisationsübergreifender Abläufe bieten. Dazu müssten die Daten und Prozesse zunächst aber erst einaml formal spezifiziert werden, um die Interoperabilität zu erreichen.

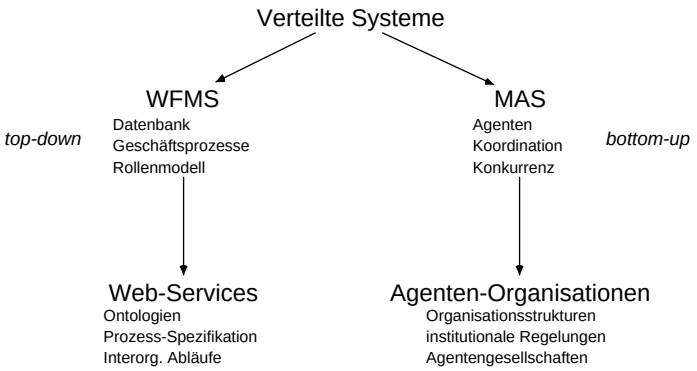


Abbildung 1.4: Konvergenz von MAS und WFMS

Die Unzulänglichkeiten der beiden Ansätze sind den Forschungsgemeinschaften wohl bewusst. So existieren Erweiterungen der WFMS um Spezifizierung der Daten und Prozesse durch Beschreibungslogiken. Diese Bemühungen werden im Bereich der *Web-Dienste* (engl. web service) und des *semantic web* gebündelt. Web-Dienste (Alonso u. a., 2003) stellen einen Standard zur Entwicklung und Integration verteilter Systeme dar. Hier existieren Technologien wie die Ontologiesprachen OWL, service oriented architecture (SOA), web service definition language (WSDL) (Gottschalk, 2000; OASIS, 1993–2007). Analog existieren Bemühungen, Koordinationsstrukturen wie Teams oder Organisationen mit leitender Wirkung im Bereich der Agentenorientierung konzeptionell zu integrieren. Diese sich abzeichnende Konvergenz von Multiagentensystemen und Workflow-Mangement-Systemen ist in Abbildung 1.4 dargestellt.

1.2 Koordination: Kooperation und Konkurrenz

Es ist im allgemeinen davon auszugehen, dass zwischen den Agenten Abhängigkeiten existieren. Funktionale Abhängigkeiten existieren, wenn ein einzelner Agent nicht alle zur Erfüllung seiner Ziele benötigten Fähigkeiten besitzt, so dass er auf die Hilfe anderer Agenten angewiesen ist. Instrumentelle Abhängigkeiten ergeben sich, sobald sich mehrere Agenten ein Hilfsmittel (z.B. ein Werkzeug) teilen, so dass es zu Synchronisationsbedingungen, beispielsweise in Form wechselseitiger Ausschlüsse, kommt. Abhängigkeiten ergeben sich auch, wenn Agenten um den Zugang zu knappen Ressourcen und deren Verwendung konkurrieren. Koordinationssituation lassen sich also anhand der drei Fragen kategorisieren:

1. Besitzt der Agent notwendige Fähigkeiten und Werkzeuge?
2. Existieren genug Ressourcen für die Zielrealisierung?
3. Sind die Ziele der Agenten kompatibel?

Die Koordination in einer Interaktionsgruppen sind idealtypischerweise entweder *kooperierend* oder *konkurrierend*.

Existiert – zumindest für den gewählten Ausschnitt – ein gemeinsames Ziel, so liegt die Situation der *Kooperation* vor. Üblicherweise sind kooperierende Interaktionsgruppen durch die Gleichartigkeit ihrer Aktivitäten gekennzeichnet, die sich aus dem gemeinsamen Ziel ergibt, auf das die Planungen der Agenten ausgerichtet sind.

Eine *konkurrierende* Interaktionsgruppe ist dadurch gekennzeichnet, dass die individuellen Ziele eines Agenten nicht gleichzeitig mit denen Zielen anderer Agenten realisiert werden können. Dies mag daran liegen, dass zwei Agenten gleichzeitig exklusiven Zugriff auf ein Betriebsmittel verlangen, oder auch daran, dass ein Verkäufer einen möglichst hohen, der Einkäufer einen möglichst niedrigen Preis anstrebt. Solche Konflikte müssen im Interesse eines reibungslosen Ablaufes aufgelöst werden. Die Lösungsansätze reichen von Algorithmen zur Deadlock-Vermeidung bis hin zu spieltheoretischen Ansätzen.

1.2.1 Kooperation: Verteiltes Problemlösen

Liegt eine gemeinsame Zielsetzung der Agenten vor, so bilden die Agenten ein *Team*, dessen Aufgaben darin besteht, die Fähigkeiten seiner Mitglieder auf das gemeinsame Ziel, die Teamaufgabe, zu koordinieren: „Wer macht wann was?“ Dieses algorithmische Problem wird als *Verteiltes Problemlösen* (engl. distributed problem solving, DPS) bezeichnet. Dieses umfasst zwei Aspekte: Zum einen die Planung und zum anderen die Ausführung. Beide Aspekte können sowohl zentralisiert als auch verteilt erfolgen. Zentralisierte Planung und Ausführung entspricht der aus der KI bekannten Problemlösen. Der Agent interagiert hier nicht mit anderen Agenten. Der Fall der zentralisierten Planung und der verteilten Ausführung kommt dann zum Einsatz, wenn die ausführenden Agenten verteilt vorliegen und mit Spezialfähigkeiten ausgestattet sind, die Planung aber nicht selbst vornehmen. Der Nachteil des Verfahrens ist, dass zur Planung ein globales Weltmodell aufgebaut werden muss. Dieser Aufwand lohnt sich aber dennoch, nämlich dann, wenn zentralisierte Algorithmen erprobter oder effizienter sind als ihre verteilten Pendants. Typische Beispiele finden sich als Optimierungsprobleme in Unternehmen, bei denen Planung und Ausführung in verschiedene Kompetenzbereiche fallen. Der umgekehrte Fall des verteilten Planens und der zentralisierten Ausführung liegt dann vor, wenn schon in die Planung die Spezialfähigkeiten der Agenten einfließt. Ein Beispiel ist hier die Konstruktion einer neuen Maschine, bei dem das Spezialwissen verschiedener Entwickler im Entwurf zusammenfließt. Existiert der fertige Entwurf, kann er in einem zentralisierten Produktionsprozess ausgeführt werden.

Der Fall des verteilten Planung und Ausführung kombiniert beide Aspekte. Hier ist also ein Plan zu erstellen, der die Aufgaben zuweist und ihre Bearbeitung synchronisiert. Bei der verteilten Planung ist zu regeln, wie die Teilaufgaben unter den Agenten entsprechend ihrer Fähigkeiten aufzuteilen sind. Dazu müssen Pläne in einer Darstellung vorliegen, die alle Agenten interpretieren können. Außerdem müssen nicht nur Aufgaben verteilt, sondern auch Teilpläne zusammengefügt und synchronisiert werden.

Die bekanntesten Protokolle zum verteilten Problemlösen sind das *Contract-Net Protocol* und das *Partial Global Planning Protocol*.

Das Contract-Net Protokoll (Smith, 1977) fokussiert auf die Aufgabenverteilung. Hier zerlegen die Manager-Agenten die Aufgabe in Teilaufgaben und schreiben sie

öffentlich aus. Auf die Ausschreibung bieten dann die Agenten. Die Aufgabe wird an den Agenten zugewiesen, der das beste Angebot abgibt. Dieser Agent, Kontraktor genannt, löst die an ihn übertragene Aufgabe und gibt die Lösung an den Manager zurück. Der Manager kombiniert abschließend die Lösungen der Teilaufgaben zu einer Gesamtlösung. Indem die Kontraktoren ihrerseits als Manager tätig werden, können sie Teilaufgaben delegieren, so dass das Contract-Net Protokoll ein verteiltes Teile-und-Herrsche Verfahren darstellt. Die selbstähnliche Struktur, bei denen Agentengruppen nach außen hin als eine Einheit agieren, sind als Holonen (Koestler, 1967) bekannt. Durch die Ausschreibung kann eine Aufgabenverteilung dynamisch festgelegt werden, die der aktuellen Verteilung von Wissen, Fähigkeit und Ressourcen der Agenten Rechnung trägt.

Das Partial Global Planning Protokoll (Durfee und Lesser, 1991) konzentriert sich stärker auf den Planungsaspekt. Es geht davon aus, dass eine Struktur existiert, die angibt, wie Teilpläne verteilt werden. Nach einer Zuweisungsphase der Aufgaben, erstellen die Agenten ihre lokalen Pläne, wobei sie dabei nur die ihnen zugewiesene Aufgabe sehen. Die lokalen Pläne enthalten dabei schon Alternativen, um die spätere Synchronisation der Pläne zu erleichtern. Die in den Teilplänen formulierten Teilziele werden von den Manager-Agenten zum Gesamtziel (engl. partial global goal) zusammengefasst. Außerdem werden die lokalen Pläne entlang des Gesamtziels ausgerichtet und miteinander synchronisiert. Der entstehende Plan bilden dann den globalen Plan (engl. partial global plan). In der nächsten Phase passen die Agenten dann ihre lokalen Pläne an den globalen an.

1.2.2 Konkurrenz: Verhandlungen

Konkurrenz liegt vor, wenn zwischen den Agenten kein Einvernehmen über das angestrebte Ziel besteht. Ohne dieses Einvernehmen stehen die Ziele der Agenten in Konkurrenz zueinander, und es liegt der Situationstyp *Wettbewerb* vor. Da Agenten in einer Konkurrenzsituation ihre Ziele so nicht realisieren können, ist im Interesse aller durch Verhandlungsprozesse ein Kompromiss zu finden.

Verhandlungsprozesse werden durch Protokolle geregelt. Bei Verhandlungsprotokollen kommt es auf die Einigungskriterien und die Kompromissmöglichkeiten an. Die grundlegende Annahme ist, dass die Akteure eigennützig handeln, sie sich also nur dann auf Verhandlungen einlassen, wenn ihnen der Kompromiss einen Vorteil bietet, wenn auch vielleicht nur einen geringeren, als wenn sie sich auf Kosten der anderen hätten durchsetzen könnten. Ein eigennütziger Akteur wird also nur dann einen Kompromiss akzeptieren, wenn das Ergebnis für ihn optimal ist, d.h. dass es bei der aktuellen Interessenlage keine bessere Entscheidung gibt.

Die Spieltheorie hat diese Form der Entscheidungsfindung zum Forschungsgegenstand. Die Grundannahme ist, dass n Agenten eigennützig ihre Ziele verfolgen und dass mit dem Erreichen eines Ziels ein Gewinn verbunden ist. Das gesamte Szenario wird als Spiel aufgefasst. Die Entscheidungsfindung jedes Agenten A_i wird durch seine Spielstrategie S_i bestimmt. Die Strategie eines Agenten orientiert sich dabei an denen der anderen. Ein Spiel der Agenten ist somit mit den Strategien $(S_1, \ldots, S_n)$ der Agenten gleichzusetzen. Beispiele für solche Spiele sind Preisbildungsprozesse, Auktionen, Wahlen oder Koalitionsverhandlungen.

Auf den Bereich der Multiagentensysteme angewendet lautet eine Teilfrage, wie Verhandlungsprotokolle – der Markt gewissermaßen – gestaltet werden müssen, damit die lokal optimierte Handlungswahl auch auf globaler Ebene ein optimales Ergebnis darstellt. Hierbei stellt sich die Frage, in welchem Verhältnis der Aufwand

des Verhandlungsprotokolls – die Spielregeln – zur Güte des Ergebnis stehen.

Aber schon allein die Formulierung des Optimalitätskriteriums gestaltet sich schwierig. Es gibt in der Literatur vier zentrale Formulierungen der Optimalität. Die erste Formulierung nennt eine Entscheidung optimal, wenn sie den kumulierten Gewinn aller Akteure maximiert. Hier wird also das Allgemeinwohl (engl. well fare) maximiert. Die zweite Formulierung nennt eine Entscheidung optimal, wenn sich kein Akteur verbessern kann, ohne dass sich mindestens ein anderer Agent verschlechtert (*Pareto-Optimalität*). Anders formuliert: Ein Agent kann sich nur auf Kosten anderer verbessern. Eine dritte Formulierung zeichnet sich dadurch aus, dass Akteure eine dominante Strategie wählen. Eine Strategie ist *dominant*, wenn sie für den Akteur optimal ist, unabhängig davon, was die anderen Akteure tun. Die wenigsten Spiele besitzen jedoch eine solche Strategie. Eine vierte Formulierung basiert auf den Begriff des Nash-Gleichgewichts. Die Strategien $(S_1, \ldots, S_n)$ sind im *Nash-Gleichgewicht*, wenn für alle A_i die Strategie S_i optimal ist, wenn die anderen Agenten nach den Strategien $(S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_n)$ spielen. Das Problem hierbei ist, dass manche Spiele kein Nash-Gleichgewicht besitzen, andere dagegen mehrere. Ein weiteres Problem ist zudem noch, dass sich selbst im Nash-Gleichgewicht Agenten konspirativ zusammenschließen können, um auf Kosten der anderen ihren Gewinn zu verbessern.

Das Verhältnis der Optimalitätskonzepte wird meist mit dem Gefangenendilemma illustriert. Beim Gefangenendilemma stehen zwei Gefangene A und B vor der Wahl, entweder zuzugeben, eine Straftat begangen zu haben oder dies abzustreiten. Streiten beide ab, so werden beide wegen geringfügiger Vergehen bestraft. Gestehen beide, so wird die Strafe aufgeteilt. Gesteht dagegen nur einer, so wird nur der leugnende Gefangene bestraft. Formulieren wir den Nutzen als die vermiedene Strafe, so ergibt sich die folgende Nutzenmatrix:

	gestehen	abstreiten
gestehen	1,1	5,0
abstreiten	0,5	3,3

An dieser Situation erkennt man, dass der allgemeine Nutzen maximiert wird, wenn beide abstreiten. Der gemeinsame Nutzen ist dann $3 + 3 = 6$. Diese Wahl ist auch Pareto-optimal, denn bei der Verbesserung zu $(5, 0)$ (bzw. zu $(0, 5)$) verschlechtert sich der Partner jeweils. Die dominante Strategie dieses Spiels ist es aber zu gestehen, denn wenn B gesteht, dann ist es für A besser, auch zu gestehen, da für ihn $(1, 1)$ besser als $(0, 5)$ ist. Streitet B dagegen ab, so ist es für A wiederum zu gestehen, da für ihn $(5, 0)$ besser als $(3, 3)$ ist. Damit ist Abstreiten für A besser, unabhängig von der Wahl von B. Wählen beide Spieler die dominante Strategie, so ergibt sich das schlechteste Ergebnis $(1, 1)$. Diese Strategiewahl ist auch das Nash-Gleichgewicht.

Die verschiedenen Optimalitätskriterien kommen also für dieses Spiel zu entgegengesetzten Strategiewahlen mit unterschiedlichem Gesamtnutzen. An dieser Stelle sei angemerkt, dass diese unterschiedlichen Ergebnisse nicht als Argument gegen die Spieltheorie oder gegen den rational-choice Ansatz verstanden werden dürfen. Vielmehr zeigt das Beispiel sehr schön, dass es Spiele gibt, bei denen lokale, egoistische Optimierung nicht zum globalen Optimum führt, sondern sogar zum denkbar schlechtesten Ergebnis. Aus der konstruktiven Perspektive besteht die Aufgabe also gerade darin, die Nutzenmatrix und damit auch die Spielregeln durch Mechanismen, also durch Ordnungspolitik, so anzupassen, dass lokale Nutzenmaximierung und Allgemeinwohl zusammenfallen.

1.3 Normative Agenten und Institutionen

Die bislang behandelten Kooperationsprotokolle gehen davon aus, dass alle Agenten eines Teams – wie im Partial Global Planning Protokoll – gutwillig an der Lösung arbeiten und die ihnen zugewiesenen Aufgaben erledigen (bene volence Eigenschaft). Da dieser Aspekt im allgemeinen zu einschränkend ist, werden für kooperative Situationen zusätzliche Sozialstrukturen berücksichtigt. Wir betrachten hier die Theorie der *joint-intentions*, bei der auch soziale Verpflichtungen der Mitglieder gegenüber der Gruppe modelliert werden.

Die Verhandlungsprotokolle besitzen den hierzu verwandten Nachteil, dass lokale Rationalität der Akteure – wie im Falle des Gefangenendilemmas – nicht zu global optimalen Lösungen führt. Ein betrachteter Ansatz ist es hier, die Rationalität mit der Theorie normativer Agenten zu kombinieren. Hierbei werden Normen, d.h. Strukturen mit verbindlichen Regelungscharakter, in den Planungsprozess der Agenten einbezogen, mit der Hoffnung, dass hierdurch die Paradoxien nicht optimaler Strategiewahlen vermieden wird.

1.3.1 Kooperationslogik

Startpunkt der Arbeiten zum *Collaborative Decision-Making (CDM)* ist die Erkenntnis, dass geteilte Ziele nicht ausreichen, um kohärentes Gruppenhandeln zu erzeugen. Daher werden die in der Gruppe geteilten Absichten zu Gruppenabsichten (engl. joint intentions) gebündelt und die einzelnen Akteure auf diese verpflichtet. Dieser Gedanke wird insbesondere in (Cohen und Levesque, 1991), (Jennings, 1993, 1996), (Castelfranchi und Conte, 1995a; Castelfranchi, 1995; Castelfranchi und Conte, 1996; Castelfranchi, 2000) und (Ossowski, 1999) vertreten. Wir betrachten im folgenden die logische Formalisierung des CDM-Prozesses nach Panzarasa u. a. (2002).

Die BDI-Modalitäten beschreiben angestrebte Weltzustände, reden aber nicht über die Prozesse, die zu ihrer Erlangung führen. Dazu bedarf es einer Prozesslogik. Die grundlegendste Struktur basiert auf der *dynamic logic* nach Harel (1984), in der Operatoren existieren, mit deren Hilfe komplexe Aktionen aus atomaren zusammengesetzt werden. Es existieren die aus der Prozessalgebra (Baeten, 1990) bekannten Konstrukte der Sequenz $(e_i; e_j)$, der alternativen Auswahl $(e_i + e_j)$, der Nebenläufigkeit $(e_i \parallel e_j)$ und der Iteration e^*. Darüberhinaus existiert der Testprozess $(\phi?)$, der blockiert, falls ϕ nicht gilt, und andernfalls keine Wirkung hat.

Wenn I eine Zeitintervall ist, dann drückt $\mathsf{Occurs}(e)(I)$ aus, dass die komplexe Aktion e in I eintritt. Die Ausführenden sind die Agenten der Gruppe gr. Hierbei wird ein Agent als einelementige Gruppe interpretiert. $\mathsf{Do}(gr, e)(I)$ drückt aus, dass die Gruppe gr die Aktion e in I ausführt:

$$\begin{aligned} \mathsf{Do}(gr, e)(I) &\equiv \mathsf{Occurs}(e) \wedge \mathsf{Agts}(gr, e)(I) \\ \mathsf{Does}(a, e)(I) &\equiv \forall gr : \mathsf{Do}(gr, e)(I) \Longrightarrow \mathsf{Singleton}(gr, a)(I) \end{aligned}$$

Der Operator plan erlaubt es, Aktionen zu repräsentieren und darüber zu schließen. Zum Zeitpunkt t ist die Aktion e für die Gruppe gr ein Plan, um zum Zeitpunkt t' den Zustand ϕ zu erreichen (mit $t < t'$):

$$\mathsf{plan}(gr, e, \phi(t'))(t)$$

Ein Plan ist durch die Aktionssequenz e charakterisiert, die geeignet ist, den Zustand ϕ zum Zeitpunkt t' realisieren, wenn sie im Zeitintervall $[t_1, t_2]$ ausgeführt

wird:

$$\begin{aligned}\mathsf{plan}(gr, e, \phi(t'))(t) \quad &\equiv \quad \exists t \leq t_1 \leq t_2 < t' : \\ &\quad \mathsf{Do}(gr, e)(t', t_2) \wedge (\mathsf{Occurs}(e)([t_1, t_2]) \Longrightarrow \mathsf{Occurs}(\phi?)(t'))\end{aligned}$$

Um Pläne abzugleichen, ist es notwendig, gemeinsame Überzeugungen zu beschreiben. Der Operator **E-Bel** beschreibt, dass zum Zeitpunkt t alle Mitglieder der Gruppe gr der Überzeugung sind, dass ϕ gilt:

$$\textbf{E-Bel}(gr, \phi)(t) \equiv \bigwedge_{a \in gr} \textbf{Bel}(a, \phi)(t)$$

Diese *allgemeine Überzeugungen* sind aber nicht reflexiv, d.h. die Agenten können gemeinsame Überzeugungen besitzen, ohne dass sie sich dessen bewusst wären. Um zu beschreiben, dass eine Gruppe von Agenten gemeinschaftliche Überzeugungen (*mutual beliefs*) besitzt, ist ein stärkerer Operator notwendig:

$$\textbf{M-Bel}(gr, \phi)(t) \equiv \textbf{E-Bel}^k(gr, \phi)(t) \text{ für alle } k$$

Dabei beschreibt $\textbf{E-Bel}^k(gr)$ das wechselseitige Wissen bis zur Stufe k:

$$\begin{aligned}\textbf{E-Bel}^0(gr, \phi)(t) \quad &\equiv \quad \phi(t) \\ \textbf{E-Bel}^{k+1}(gr, \phi)(t) \quad &\equiv \quad \textbf{E-Bel}(gr, \textbf{E-Bel}^k(gr, \phi)(t))(t)\end{aligned}$$

Analog zu **E-Bel** werden auch allgemeine Bedürfnisse (**E-Des**), Ziele (**E-Goal**) und Absichten (**E-Int**) definiert.

Eine Gruppe gr besitzt eine *gemeinschaftliches* Bedürfnis (engl. joint intention) ϕ, notiert: $\textbf{J-Des}(gr, \phi)(t)$, wenn folgendes gilt:

1. Jeder Agent $a \in gr$ hat ein Bedürfnis bezüglich ϕ.
2. Es besteht gemeinschaftliche Überzeugung, dass jeder Agent ein Bedürfnis bezüglich ϕ hat.
3. Jeder Agent hat die Absicht, dass alle anderen Agenten aus gr ein Bedürfnis bezüglich ϕ haben.
4. Es besteht gemeinschaftlich Überzeugung über den vorherigen Punkt.

Diese vier Bedingungen lassen sich folgendermaßen formalisieren:

$$\begin{aligned}\textbf{J-Des}(gr, \phi)(t) \quad &\equiv \quad \textbf{E-Des}(gr, \phi)(t) \\ &\quad \wedge \quad \textbf{M-Bel}(gr, \psi)(t) \\ &\quad \wedge \quad \textbf{E-Int}(gr, \psi)(t) \\ &\quad \wedge \quad \textbf{M-Bel}(\textbf{E-Int}(gr, \psi))(t) \\ &\quad \quad \text{mit } \psi = \textbf{E-Des}(gr, \phi)(t)\end{aligned}$$

Die Operatoren **J-Goal** und **J-Int** für gemeinschaftliche Ziele und Absichten werden analog definiert.

Gemeinschaftliche Bedürfnisse reichen nicht aus, den Grad der Verbindlichkeit innerhalb der Gruppe zu gewährleisten, der notwendig ist, um eine gemeinsame Aktion auszuführen. Als weiteres notwendiges Konstrukt ist das Konzept der *Verpflichtung*

(engl. commitment) zu sehen. Verpflichtungen bündeln die von Gruppen geteilte Absichten (engl. joint intentions) in einem überindividuellen Element. Als zusätzliche Persistenz-Bedingung zum Sicherstellen gemeinschaftlichen Handelns dienen *soziale Verpflichtung* (engl. social commitments), mit denen sich die Gruppe gr_1 gegenüber der Gruppe gr_2 zur Ausführung der Aktionssequenz e verpflichtet:

$$\mathsf{Comm}(gr_1, gr_2, e)(t)$$

Analog wird durch $\mathsf{Comm}(gr_1, gr_2, \psi(t'))(t)$ beschrieben, dass sich die Gruppe gr_1 zum Zeitpunkt t gegenüber gr_2 verpflichtet, zum Zeitpunkt t' den Zustand ψ herzustellen. Gemeinschaftliche Absichten heißen persistent (engl. persistent joint intentions), falls sie durch soziale Verpflichtungen entsprechend gestärkt sind.

Hiermit wird das Konzept der gemeinsamen sozialen Verpflichtung (engl. *joint commitment*) entwickelt. Zum Zeitpunkt t besitzt die Gruppe gr eine gemeinsame soziale Verpflichtung bezüglich des Erreichens von ϕ zum Zeitpunkt t', wenn es für gr eine persistente gemeinsame Absicht bezüglich $\phi(t')$ gibt.

1. Die Gruppe gr ist der Überzeugung, dass ϕ zum Zeitpunkt t' wahr sein wird.
2. gr besitzt eine gemeinsame Absicht, dass ϕ zum Zeitpunkt t' wahr sein wird.
3. Jeder Agent $a \in gr$ hat sich sozial gegenüber der Gruppe gr bezüglich dieser gemeinsamen Absichten verpflichtet.
4. In der Gruppe gr besteht gemeinschaftliche Überzeugung bezüglich (3).
5. Es ist wahr und gemeinschaftliche Überzeugung, dass (2) solange gilt, bis entweder gr gemeinschaftlich der Überzeugung ist, dass ϕ zum Zeitpunkt t' falsch sein wird, oder mindestens ein Agent die Gruppe verlässt.

Der fünfte Punkt beschreibt die einzige Möglichkeit, einer sozialen Verpflichtung nicht nachzukommen. Formal wird eine soziale Verpflichtung durch die fünf Konjunkte des folgenden Ausdrucks beschrieben:

$$\begin{aligned}
\textbf{J-Comm}(gr, \phi(t'))(t) \quad \equiv \quad & \textbf{M-Bel}(gr, \phi(t'))(t) \wedge \\
& \textbf{J-Int}(gr, \phi(t'))(t) \wedge \\
& \bigwedge_{a \in gr} \Big(\mathsf{Comm}(a, gr, \phi(t')) \\
& \qquad \wedge \textbf{M-Bel}(gr, \mathsf{Comm}(a, gr, \phi(t'))) \Big)(t) \wedge \\
& \gamma(t) \wedge \textbf{M-Bel}(gr, \gamma))(t)
\end{aligned}$$

Die Formel γ drückt dabei den fünften Punkt aus: Entweder ϕ ist gemeinschaftliche Überzeugung oder die Gruppe gibt diese zum Zeitpunkt t_2 auf. Dies ist entweder der Fall, wenn die Gruppe gemeinschaftlich der Überzeugung ist, dass ϕ zum Zeitpunkt t' falsch sein wird, oder wenn mindestens ein Agent a seiner sozialen Verpflichtung nicht nachkommt. Die gemeinsame Absicht gilt aber dann bis zum Zeitpunkt t_2:

$$\begin{aligned}
\gamma \equiv \; & \textbf{J-Int}(gr, \phi(t'))(t) \vee \\
& \exists t < t_2 < t' : \Big[\textbf{M-Bel}(gr, \neg\phi(t'))(t) \\
& \qquad \vee \exists a \in gr : \big(\neg \mathsf{Comm}(a, gr, \phi(t')) \\
& \qquad\qquad \wedge \textbf{M-Bel}(gr, \neg \mathsf{Comm}(a, gr, \phi(t'))) \big) \Big](t_2) \\
& \wedge \quad \forall t < t_1 < t_2 : \textbf{J-Int}(gr, \phi(t'))(t_1)
\end{aligned}$$

Besteht eine soziale Verpflichtung **J-Comm**$(gr, \phi(t'))(t)$, dann können alle Gruppenagenten sicher sein, dass kein Mitglied seine Absichten spontan ändert, ohne dass dazu Gründe vorliegen, die die gemeinsamen Absichten tangieren. Auf diese Art und Weise wird eine größere Verbindlichkeit und damit auch eine größere Planungssicherheit erreicht.

1.3.2 Normativ handelnde Agenten

Das Konzept der Norm wurde eingeführt, um zwischen der lokalen, d.h. individuellen Rationalität der Agenten und den global erwünschten Prozessen eine Verbindung herzustellen (für eine soziologische Perspektive zum Nutzen der Norm für kooperative Agenten siehe auch Saam, 2001). Diese Abkehr von der Theorie der rationalen Handlungswahl (engl. rational choice) stellt zunächst einmal die Grundannahme, dass Agenten im Rahmen ihrer Logik rational handeln, in Frage.

Als Gegenentwurf zum rational handelnden *homo oeconomicus* dient hierzu der *homo sociologicus*, der in seiner Handlungswahl nicht von einer Kosten-Nutzen Abwägung, sondern von normativen Kategorien geleitet wird . Auch die Wirtschaftswissenschaften teilen mittlerweile diese Auffassung, da die Empirie zeigt, dass die Akteure des Marktes in der Realität sich nicht rational im Sinne einer Gewinnmaximierung verhalten.

Ullman-Margalit (1977) gibt verschiedene rationale Funktionen für Normen an. Erstens verbessern Normen die Kooperation zwischen Agenten, indem sie lokal den Suchraum beschränken. Normen dienen dann der Komplexitätsreduktion. Zweitens verbessern Normen in sozialen Dilemmata die Strategiewahl, indem sie lokal rationales, aber global unerwünschtes Verhalten ausblenden und so den Gesamtnutzen verbessern. Die Wirkung von Normen wurde in Simulationsversuchen exemplarisch bestätigt (vgl. Shoham und Tennenholtz, 1994; Walker und Wooldridge, 1995).

Für die Modellierung stellt sich die Frage, wie das Verhältnis von normativem und rationalen Handeln zu fassen ist. Intuitiv erwarten wir, dass sich normative Agenten anders verhalten sollten als langfristige planende Kosten-Nutzen-Maximierer. Man kann sogar fragen, ob beide Konzepte – Rationalität und Normativität – miteinander vereinbaren lassen oder ob sie einander ausschließen. Dies ist identisch mit der Frage, ob Normen überhaupt im Rahmen der planenden Handlungswahl formalisierbar sind, denn sobald Normen Bestandteil der Planung sind, verschwindet der Gegensatz zwischen rationalen und normativen Handeln, da dann normatives Verhalten rational ist. Beispielsweise mag ein Verhalten, das auf kurze Sicht nicht optimal im Sinne von Kosten-Nutzen Erwägungen ist, dies sehr wohl aber für einen größeren Betrachtungszeitraum sein. Als Konsequenz ergibt sich Normativität – obgleich sie von den Akteuren gar nicht berücksichtigt oder angestrebt wird – von allein, quasi durch die „unsichtbare Hand des Marktes“ individueller Nutzenmaximierung.

Derartige Überlegungen geben uns leider keinerlei Hinweis auf die Strukturen, mit denen die Handlungswahl erfolgt, denn logisch gesehen kann die Beeinflussung von Rationalität und Normativität sogar umgekehrt sein, d.h. die rationale Handlungswahl kann Nebenprodukt normativ geleiteter Prozesse sein. Rationalität und Normativität können sogar von einem Mechanismus erzeugt werden, der keinerlei direkten bezug zu beiden Aspekten aufweist. Die Frage ist daher nicht, welche die „richtige“ Struktur ist, sondern welche besonders beschreibungsökonomisch ist. Von dieser Prämisse ausgehend ist es sinnvoll, Normen als Teil der Handlungswahl direkt zu breücksichtigen. Konsequenterweise betrachten Castelfranchi und Conte

(1995c,b,a) Agenten, deren Planen von normativen Elementen bestimmt wird. Um normkonformes Handel programmieren zu können, sind Normen geeignet zu repräsentieren. Hierzu sind mehrere Modellierungsformen denkbar. Castelfranchi und Conte (1995b) schlagen folgende Varianten vor:

1. Normen sind als Randbedingungen (engl. constraints) formuliert, die jeder Planungsprozess zu erfüllen hat. Normen sind somit der Planung an sich entzogen. Hierbei bleibt zunächst offen, wie die Dynamik der Normen zu konzeptionalisieren ist.

2. Normen sind spezielle Ziele eines Agenten, sie werden also von der Planung angestrebt. Agentenkonzepte, die diesem Ansatz folgen, operieren mit einem differenzieren Begriff eines *Ziels*. Ziele können individuellen Charakter besitzen, sie können in einer Agentengruppe geteilt werden, oder sie können sogar normativer Natur sein.

3. Normen sind eigenständige Objekte des Planungsprozess, die sich insbesondere von Zielen unterschieden. Es gibt normative Überzeugungen und Regeln, nach denen eine Norm für einen Agenten handlungsleitend wird.

Der erste Ansatz hat zur Konsequenz, dass es Agenten nicht möglich ist, gegen Normen zu verstoßen, was im zweiten und dritten Fall sehr wohl möglich ist, da ein Agent Ziele bzw. normative Erwartungen gegeneinander abwägen kann. Auch ist es nicht möglich, dass sich Normen verändern, da der Agent ihm Rahmen seiner Planung keinerlei Zugriff auf die Darstellung der Randbedingungen nehmen kann. Normen können so nicht emergieren oder sich über die Zeit verändern. Sie sind programmiert.

Verglichen mit dem dritten ist das Normkonzept des zweiten Ansatzes sehr schwach, denn Normen werden stets den normalen Zielen untergeordnet, sobald sich deren Verfolgung als lohnender darstellt. Normen haben also im Extremfall keinerlei Verbindlichkeit für den Agenten.

Castelfranchi und Conte (1995b) argumentieren daher für den dritten Ansatz, der den Normen eine eigenständige Rolle zuweist. Eine Architektur, die dem dritten Ansatz folgt, wird in (Castelfranchi u. a., 1999) vorgestellt. Dieser Ansatz ist sehr flexibel, da er Deduktion über dem Normensystem erlaubt. Die generelle Frage ist für diesen Ansatz, unter welchen Umständen ein Agent seine normative Überzeugungen sich zu eigen macht (engl. norm adoption).

Wir betrachten die Formalisierung nach (Castelfranchi und Conte, 1999). Normative Annahmen eines Agenten a über die Aktion e für die Gruppe gr sind als die Verpflichtung aller Gruppenmitglieder, die Aktion e auszuführen, formalisiert (engl. normative believe):

$$\mathbf{N\text{-}Bel}(a, gr, e) \equiv \bigwedge_{a' \in gr} \mathbf{Bel}(a, \mathsf{Ought}(\mathsf{Does}(a', e)))$$

Ein Agent glaubt an die Relevanz einer Norm für sich, wenn er glaubt, zur Agentengruppe, die von der Norm angesprochen wird, zu gehören (engl. normative believe of pertinence):

$$\mathbf{P\text{-}N\text{-}Bel}(a, e) \equiv \mathbf{N\text{-}Bel}(a, gr, e) \wedge \mathbf{Bel}(a, a \in gr)$$

Normative Ziele werden mit Hilfe der Modalität der relativen Ziele formalisiert. Hierbei bedeutet $\mathbf{R\text{-}Goal}(a, \phi, \psi)$, dass der Agent a das Ziel ϕ verfolgt, solange er

glaubt, dass ψ gültig ist. Damit ergibt sich die Formalisierung normativer Ziele:

$$\textbf{N-Goal}(a, e) \equiv \textbf{R-Goal}(a, \mathsf{Does}(a, e), \textbf{P-N-Bel}(a, e))$$

Dies bedeutet, dass der Agent a das normative Ziel bezüglich e besitzt, wenn er das Ziel hat, e auszuführen, und dies solange er sich dazu normativ verpflichtet fühlt.

Zusammenfassend kann man sagen, dass sich dieser Ansatz durch eine explizite Repräsentation normativer Verpflichtungen auszeichnet. Die Darstellung der Normen ist dabei klar getrennt von denen der Ziele. Normen und Ziele sind im Rahmen des Planungprozesses miteinander verkoppelt, bei dem der Agent selbst entscheidet, wann er sich welche Normen zu eigen macht.

1.4 Multiagentensystemorganisationen

Bislang haben wir Agentensystemen so betrachtet, als ob die Agenten egalitär wären: Jeder Agent kann mit jedem kommunizieren, d.h. dass das soziale Netzwerk zwischen den Agenten vollständig vermascht ist. Es wird meist implizit davon ausgegangen, dass die relevanten Agenten einander bekannt sind, bzw. einander anhand eines Namensdienstes leicht auffinden können.

Dieser Ansatz ist offensichtlich nur für eine geringe Anzahl von Agenten praktikabel. Überschreitet die Agentenpopulation eine bestimmte Größe, so ist eine explizite Interaktion aller mit allen parktisch nicht mehr möglich. Vielmehr ist eine Strukturierung notwendig. Außerdem hat diese Darstellung, den Nachteil, dass das Agentensystem kein von den Agenten unabhängiges Gedächtnis besitzt, da Agenten einander stets nur als Individuen begegnen. Ein strukturelles Gedächtnis – kodiert in Erwartungen, Rollen, Positionen, Netzwerke usw. – hilft im allgemeinen, die eigentlichen Interaktionen zu flankieren. Mit Hilfe des strukturellen Gedächtnisses können Agenten darauf bauen, dass andere ihren Rollen gerecht werden. Dies ist wichtig für Situationen, die Erwartungssicherheit verlangen. Agenten bringen also in erster Linie der Rolle und erst in zweiter Linie ihrem Träger Vertrauen entgegen. Ohne strukturelles Gedächtnis können Agenten Vertrauen dagegen nur Individuen entgegenbringen und sind somit gezwungen, dieses stets individuell neu aufzubauen.

Eine solche Systemstruktur drückt sich im Konzept der MAS-Organisation aus (siehe dazu Ferber u. a., 2003; dos Reis Coutinho u. a., 2005). Es mag auf den ersten Blick verwundern, dass an dieser Stelle der Systemstruktur ähnliche Attribute zugesprochen werden wie einem Agenten – speziell Intelligenz und Lernfähigkeit. Aber bereits Müller (1993) geht von dem VKI-Konzept der *organisationellen Intelligenz* aus. Organisationen verfügen hiernach über Kenntnis der Organisationsziele, über Wissen der Handlungsalternativen, über die Fähigkeit, die jeweils beste Handlungsalternative zu erkennen, auszuwählen und zu verfolgen, über Lernfähigkeit und sogar über ein organisationelles Gedächtnis. Das Petrinetz in Abbildung 1.5 illustriert die elementaren Beziehungen einer Organisation, die mittels ihrer Organisationsmitglieder sowohl die Organisationsziele koordinativ umsetzt als auch die Organisationsparameter überwacht und mit den Zielen rückkoppelt.

Obwohl das Konzept der Organisation als überindividuelles Konstrukt schon länger in der VKI diskutiert wird, bildet sich eine Organisationstheorie der Multiagentensysteme gerade erst unter dem Stichwort der *Computational Organsisation Theory* heraus (vgl. dazu Prietula u. a., 1998; Carley und Gasser, 1999). Diese Theorie hat ihre Wurzeln sowohl in der VKI als auch im Operations Research und der theoretischen Betriebssoziologie.

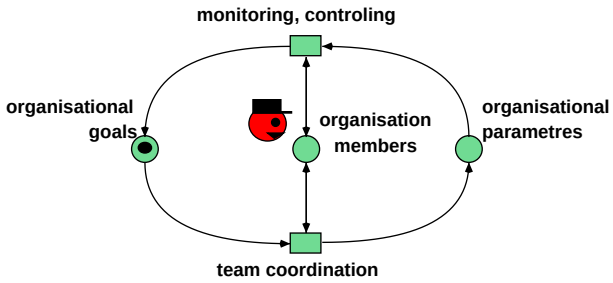


Abbildung 1.5: Multiagentensystemorganisation

Organisationen dienen dazu, die Einschränkungen von Akteuren zu überwinden. Die größte Einschränkung ist dabei die Tatsache, dass Akteure als beschränkt-rational anzusehen sind, d.h. ihre Handlungswahl ist geprägt von unvollständiger Information und beschränkten Ressourcen (Zeit, Geld etc.). Organisationen sollen die resultierende Suboptimalität kompensieren, indem sie beispielsweise Mechanismen entwickeln, die definieren, welche Informationen aus der Umwelt als wichtig zu erachten sind, und die definieren, welche Substrukturen der Organisation auf die relevanten Informationen zu reagieren haben. Zudem wird noch definiert, wie dies geschehen soll, d.h. die Abläufe und die daran beteiligten Akteure sind festgelegt. Organisationen bilden also einen ordnenden Rahmen für die Handlungen der beteiligten Agenten, wobei meist unterstellt wird, dass – zumindest implizit – die Kriterien, nach denen diese Rahmung erfolgt, zielgeleitet sind. Dieser Rahmen dient u.a. dazu, die Unsicherheit der Umwelt zu begrenzen und damit auch den Koordinierungsaufwand und die Entscheidungskomplexität zu reduzieren – idealerweise zu minimieren.

Kurzgesagt sind Organisationen somit vorstrukturierte Systeme, in denen sich Rollen, Normen etc. konzentrieren. Organisationsformen dienen dabei auf der Meso-Ebene der Strukturierung von Interaktion zwischen Organisationsmitglieder und auf der Makro-Ebene der Koordination zwischen verschiedenen Organisationen (vgl. Schillo und Spresny, 2005).

Die konkrete Ausgestaltung einer Organisation zu einer gegebenen Aufgabe hängt von verschiedenen Faktoren ab: Zum einen natürlich von der Aufgabe selbst, zum anderen aber auch von dem Wissen und den Fähigkeiten der Akteure. Daneben spielt die Dynamik der Umgebung eine große Rollen, denn die notwendigen Reaktionszeiten schließen von vorherein gewisse Organisationsstrukturen aus. Nicht zuletzt spielt noch eine Rolle, welches Optimalitätskriterium man heranzieht. Hier kann je nach Kontext auf maximale Effizienz, geringes Risiko, minimale Kosten, maximalen Gewinn oder optimale Qualität Wert gelegt werden.

Folgt man Carley und Gasser (1999), dann existieren viel zentrale Organisationsperspektiven: Die Perspektive der Aufgaben, der Akteure, der Strukturen und der verfügbaren Technologien.[4]

Die *Aufgaben*, denen sich eine Organisation stellt, können nach verschiedenen Parametern klassifiziert werden. Beispiele für solche Parameter sind der Koordinierungsbedarf (Ist die Aufgabe isoliert zu erledigen oder ist sie in einen Gesamtkontext eingebunden?), der Grad der Wiederholung (Handelt es sich um eine Routineaufgabe oder um neuartige Anforderung?), die Dynamik der Umgebung (Ist die

[4] Wir geben an dieser Stelle nur einen kurzen Einblick in die Organisationstheorie.

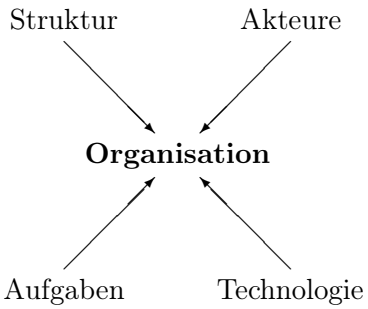


Abbildung 1.6: Die Organsiationsperspektiven nach Carley und Gasser (1999)

Umgebung während der Bearbeitung weitesgehend stabil oder verändern sich die Anforderungen laufend?), die Komplexität (Sind einige wenige oder sehr viele Parameter zur Bearbeitung heranzuziehen?), der Grad der Spezialisierung (Sind nur wenige oder sehr viele verschiedene Fähigkeiten zur Bearbeitung notwendig?) und der Ressourcenbedarf (Ist der Bedarf hoch oder niedrig?). Die konkrete Zusammensetzung aller Parametersätze und ihre zu erwartende Entwicklung über die Zeit bestimmt, welche Organisationsformen sich positiv auf die potentielle Bearbeitbarkeit auswirken.

Die *Organisationsstrukturen* definieren sich klassischerweise anhand eines Netzwerkes. In Unternehmen beschreiben diese eine Struktur zwischen Positionen. Es existieren verschiedene Erscheinungsformen der Organisation (Horling und Lesser, 2005a). Die bekanntesten Formen sind:

- Hierarchien: Die Agenten kommunizieren nur mit ihren Nachbarn in der Hierarchie, die eine baumartige Struktur besitzt. Typischerweise verläuft die Delegation in Richtung der Blätter, während die Verantwortlichkeit auf die Wurzel zielt.

- Koalitionen: Agenten schließen sich zu Koalitionen zusammen, um den individuellen Nutzen der Mitglieder zu steigern. Koalitionen treten nach außen als eine Einheit auf. Der Zusammenschluß ist auf das Fortbestehen des gemeinsamen Nutzens beschränkt.

- Teams: Agenten schließen sich zu Teams zusammen, um eine gemeinsame Aufgabe kooperativ lösen zu können. Ähnlich zu Koalitionen ist ein Team in seinem Bestehen auf die Dauer des gemeinsamen Ziels beschränkt.

- Kongregationen: Hierbei handelt es sich um einen Zusammenschluß, der nicht auf einen konkreten Nutzen oder ein konkretes Ziel ausgerichtet ist. Stattdessen schließen sich hier Agenten mit ähnlichen Fähigkeiten und Interessen zu langfristig operierenden Einheiten zusammen.

- Föderation: Hierbei schließen sich Agenten zu Teilgruppen zusammen. Jede Teilgruppe wird von einem Delegierten (engl. mediator) gegenüber den anderen Delegierten vertreten.

- Holonische Organisation: Holonen sind rekursiv geschachtelte Strukturen, die nach außen als ein Agent betrachtet werden. Sie sind somit sowohl mit dem hierarchischen Organisationstyp als auch mit der Föderation verwandt.

- Matrix-Organisationen: Hier wird die Aggregationsbeziehung flexibilisiert. Während bei Hierarchien, Holonen oder Föderation jeder Agent in einer Einheit aggregiert ist, gibt es in der Matrix mehrere Vertreter.

- Märkte: In elektronischen Märkten organisieren sich Anbieter und Kaufinteressierte. Im Gegensatz zu anderen Formen stehen die Akteure eines Marktes stets in Konkurrenz zueinander.

- Gesellschaften: Es finden sich Agenten mit heterogenen Zielen und Fähigkeiten zusammen und entwickeln dabei eigene soziale Konventionen und Normen.

Die Netzwerke legen zumeist die Verantwortlichkeiten und Kompetenzen zwischen Positionen fest, beispielsweise die Weisungs- oder die Delegationsbefugnis. Die Knoten dieses Netzwerks bilden hierbei die *Positionen*, die definieren, welche Rollen der Positionsinhaber einnehmen kann bzw. muss, welche Ressourcen er mobilisieren kann usw. Eine *Rolle* definiert situationsbezogen, welche Aktionen ausgeführt werden dürfen bzw. welche ausgeführt werden müssen, d.h. die *Rechte* und *Pflichten* einer Rolle.

Die Akteure einer Organisation sind nun als Organisationsmitglieder die Inhaber der *Positionen*. Jeder Akteure besitzt Wissen und Fähigkeiten, im Idealfall mindestens jene, die zur Ausübung der mit seiner Position verbundenen Rollen notwendig sind. Bedingt durch seine lokale Einbettung in den Organisationszusammenhang besitzt ein Akteur im allgemeinen nur eine eingeschränkte Sicht auf die Organisation, woraus sich Divergenzen von organisational intendierten und empirisch beobachtbaren Verhalten erklären lässt.

Technologien sind von der Umwelt bereitgestellte Mechanismen, die sich eine Organisation prinzipiell nutzbar machen kann, vorausgesetzt ihre Akteure sind in der Lage, diese zu verwenden. Komplexe Maschinen sind in diesem Sinne eine Technologie, denn sie stehen im Prinzip jeder Organisation zur Verfügung, gleichzeitig ist aber offen, ob die Organisationsmitglieder sie bedienen können. Technologien bestimmen somit, wie organisationale Absichten tatsächlich operationalisiert werden, welche Ressourcen dazu jeweils notwendig sind und welche Fähigkeiten auf seiten der Akteure notwendig sind.

Die Abhängigkeiten der Konzepte sind in Abbildung 1.7 dargestellt. Die Elemente, die der formalen Organisation zuzuordnen sind (Netzwerk, Position, Rolle etc.) sind gegenüber den informalen, externen Elemente (Akteure, ihre Fähigkeiten, Technologien und die Aufgaben der Organisation) hervorgehoben.

1.4.1 Offene Plattformen als Organisationen

Organisationen spielen insbesonders für *offene Agentensysteme* eine zentrale Rolle. Offene Agentensysteme zeichnen sich durch eine dynamische Agentenpopulation aus, die sich ergibt, indem die vorhandenen Agenten das System verlassen und neue es betreten. Bekanntestes Beispiel hierfür sind Plattformen aus dem Bereich des elektronischen Handels (engl. electronic commerce) (vgl. Abbildung 1.8): Die Aufgabe einer Handelsplattform ist es, Verkäufer und Käufer zusammenzubringen und sie bei der Preisverhandlung sowie der Vertragsabwicklung zu unterstützen. Die konkreten Verkäufer- und Käuferagenten (hier: $V_1, \ldots, V_m$ und $K_1, \ldots, K_n$) variieren über die Zeit hinweg. Als weitere Besonderheit offener Systeme kommt hinzu, dass diese Agenten extern entwickelt werden. Dies hat für die Entwicklung einer Handelsplattform zur Konsequenz, dass es nicht möglich ist, das Verhalten

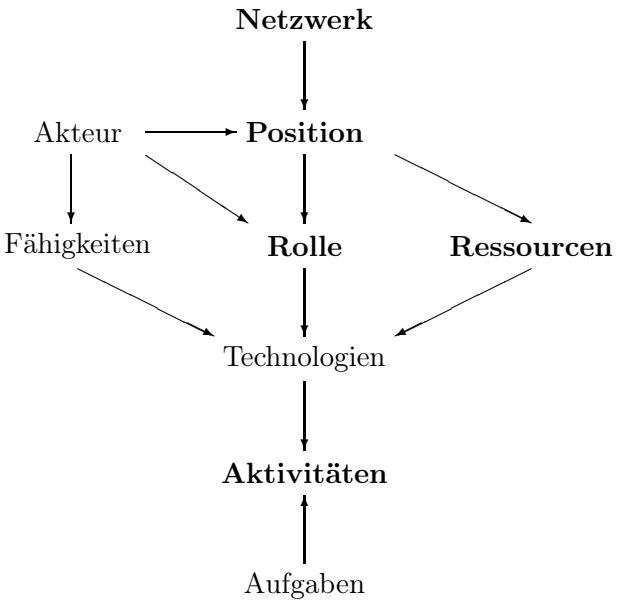


Abbildung 1.7: Die Relationen zwischen Organisationskonzepten

der auf der Plattform agierenden Agenten vorab adäquat zu gestalten. Um den ordnungsgemäßen Ablauf zu gewährleisten ist es jedoch unumgänglich, das Verhalten der Agenten geeignet einzuschränken. Da dies aufgrund der Autonomie der Agenten nicht möglich ist, werden von der Plattform stattdessen Rollen und Interaktionsprotokolle (hier: *Verkäufer* und *Käufer*) spezifiziert und die Agenten auf das so definierte Verhalten festgelegt – beispielsweise durch Kontroll- und Sanktionierungsmechanismen.

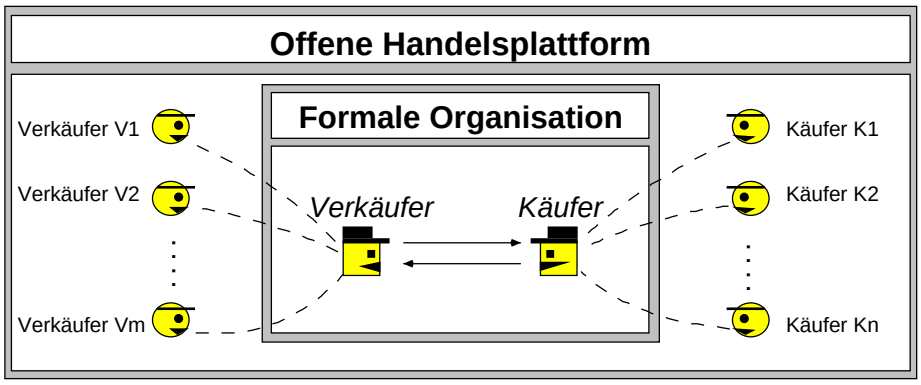


Abbildung 1.8: Eine offene Handelsplattform

Hier sieht man sofort, dass für die Entwicklung einer Handelsplattform überindividuelle Konstrukte von großem Nutzen sind. Die Handelsplattform ist somit das Paradebeispiel einer formalen Organisation (siehe dazu auch Kapitel **??**), da die strukturell getrennte Entwicklung von Plattform einerseits und Käufer- und Verkäuferagenten andererseits eine analytische Aufspaltung des Systems in den formalen und den informellen Anteil notwendig macht.

1.4.2 Agentenorientierte Softwareentwicklung

Die agentenorientierte Softwareentwicklung (AOSE) betrachtet das Agentenkonzept als Fortführung des Objektkonzeptes. Jennings (2000) argumentiert, dass die agentenorientierte Softwareentwicklung genau die Aspekte bereitstellt, die Booch (1992) als zentrale Mechanismen zur Bewältigung der Komplexität moderner Software ansieht:

1. Abstraktion: Es wird nicht das komplette Modell erstellt, sondern zu verschiedenen Phasen werden nur ausgesuchte Aspekte modelliert. Hier unterstützt das Autonomiekonzept den Entwickler, indem es ihm erlaubt, Agenten so zu implementieren, dass sie nur auf eine Teilmenge der Kommunikationen eingehen.

2. Dekomposition: Das System wird so lange in Subsysteme zerlegt, bis diese eine beherrschbare Größe erreichen. Die Agentenmetapher unterstützt diese Trennung, indem sie bereits Hinweise bereit hält, entlang welcher Linien die Trennung zu erfolgen hat, nämlich entlang von Ressourcen, Wissen, Fähigkeiten usw.

3. Hierarchie/Organisation: Hierbei geht es darum, wie die Subsysteme in einen funktionalen Zusammenhang integriert werden können. Die Modellierung der MAS-Organisation stellt sich genau diese Frage und analysiert die Interaktion in Bezug auf Rollen, Positionen und Kommunikationsstrukturen.

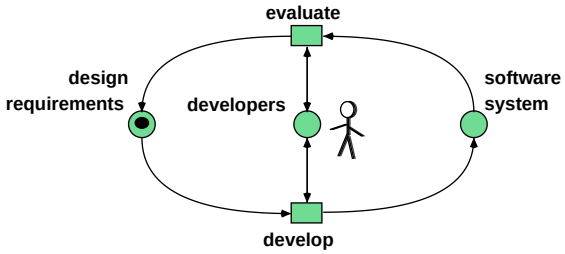


Abbildung 1.9: Iteratives Entwicklungsmodell

Im Rahmen der Softwareentwicklung bilden Entwicklungsmodelle, die auf dem iterativen Prozess der Abbildung 1.9 basieren, eine hervorgehobene Stellung. Im Rahmen der agentenorientierten Softwareentwicklung haben sich zahlreiche Spezialisierungen dieser Entwicklungsmodelle herausgebildet, beispielsweise MASE (DeLoach u. a., 2001), GAIA (Zambonelli u. a., 2003), TROPOS (Bresciani u. a., 2004), OMNI (Dignum u. a., 2004), MOISE (Hannoun u. a., 2000) und ODML (Horling und Lesser, 2005b), um einige zu nennen. Beim der agentenorientierten Softwareentwicklung ist festzulegen, wie Agenten modelliert werden, d.h. welche Ontologie verwendet wird, wie ihr Wissen und ihre Ziele modelliert werden und mit welchen Interaktionsprotokollen sie in welchen Rollen mit anderen Agenten interagieren. Bei dieser Modellierung liegt der Fokus auf den Agenten des Systems. Betrachtet man das Multiagentensystem dagegen unter dem Fokus des Systems, so stellt man fest, dass mit der Analyse der Rollen und der Agenten-Interaktionen die Systemstruktur noch nicht festgelegt ist. Es sind noch die Organisationsstrukturen (Positionen, Netzwerke usw.) zu definieren, in die Agenteninteraktionen eingebettet sind. Wir unterscheiden im folgenden zwischen der funktionalen Perspektive, die sich in ihrer

Natur auf die statische Typisierung von Agenten bezieht, und der Strukturen, die sich auf die Interaktionsnetzwerke bezieht.

Funktionale Typisierung Eine offene Frage ist, welchen Grad der Spezialisierung und der Redundanz die einzelnen Agenten aufweisen sollen. Die Spezialisierung ist hierbei ein qualitativer Aspekt, die Redundanz ein quantitativer. Diese Fragestellung spricht das statische Design eines Multiagentensystems an. Ferber (1999) spricht in diesem Zusammenhang von der *funktioneller Analyse* der MAS-Organisation. Die einander – zum Teil einander entgegengerichteten – Anforderungen lassen sich folgendermaßen beschreiben:

- Die Spezialisierung soll einerseits die größtmögliche Aufgabenfokussierung leisten und
- andererseits mit dem kleinstmöglichen Koordinierungsaufwand auskommen.
- Die Redundanz soll einerseits den größtmögliche Durchsatz erlauben und
- andererseits mit dem kleinstmöglichen Ressourceneinsatz auskommen.

Das eine Extrem bezüglich der Spezialisierung ist der omni-potente Agent, der alle Rollen realisiert. Er hat einen sehr geringen Koordinationsoverhead, da alle Kommunikation intern realisiert wird. Er ist allerdings sehr komplex und daher schwer zu warten. Das andere Extrem ist die Spezialisierung von Agenten auf genau eine Rolle, was die größtmögliche Trennung von Aufgaben bedeutet, aber meist auch einen maximalen Koordinierungsaufwand.

Auch hier ergeben sich im allgemeinen entgegengesetzte Anforderungen: Die Rollen, die potentielle Engpässe darstellen, weil sie besonders nachgefragt sind oder ihr Ausfall kritisch wäre, sollten redundant ausgelegt sein. Dies kann bedeuten, dass mehrere Agenten sich die Aufgabenlast teilen, oder dass ein Agent im Notfall für einen ausgefallenen Agenten „einspringen“ kann. Im allgemeinen sind die Ressourcen jedoch begrenzt, so dass nur ein begrenztes Maß an Redundanz realisiert werden kann.

Interaktionsstrukturen Neben dem statischen Design ist weiterhin die Dynamik des Systems zu entwerfen. Noch offene Designentscheidungen betrffen beispielsweise die Kommunikationsinfrastruktur des Systems, d.h. wer mit wem kommunizieren kann, welche Agenten wem gegenüber Arbeit delegieren dürfen usw. Ferber (1999) spricht hier von *struktureller Analyse*. Die Anforderungen lassen sich folgendermaßen beschreiben:

- Die Systemstruktur soll die größtmögliche Flexibilität in Hinblick auf die Aufgabenbewältigung bieten.
- Sie soll mit dem kleinstmöglichen Aufwand an Ressourcen (Nachrichtenkomplexität etc.) auskommen.

Dies lässt sich an einigen Extremfällen illustrieren. Eine Teamstruktur mag sich dadurch auszeichnen, dass keine Hierarchien existieren, jeder Agent jeden kennt und keinerlei Weisungs- oder Delegationstrukturen existieren. Der Vorteil besteht in der Flexibilität, denn alle Agenten können potentiell zur Aufgabenbewältigung beitragen. Dies ist gleichzeitig auch der Nachteil, denn für große Anzahlen von Agenten

wird ein nicht unwesentlicher Anteil an Ressourcen dafür verwendet, die geeigneten Kommunikationspartner zu ermitteln. Das andere Ende des Spektrum stellt eine totale Bürokratie dar. In einer Bürokratie existieren Hierarchien, an denen sich auch die Weisungsmöglichkeiten orientieren. Bekanntschaft ist nur zwischen benachbarten Hierarchieebenen nötig, weitere Interaktionsbeziehungen sind nicht vorgesehen und somit in formaler Hinsicht auch ausgeschlossen. Die Kommunikationsstrukturen erleichtern das Auffinden der zuständigen Agenten, jedoch um den Preis, dass viele Bewältigungsmöglichkeiten von vornherein ausgeschlossen werden. Es ergeben sich also entgegensätzliche Forderungen an Flexibilität und Koordinierungsaufwand.

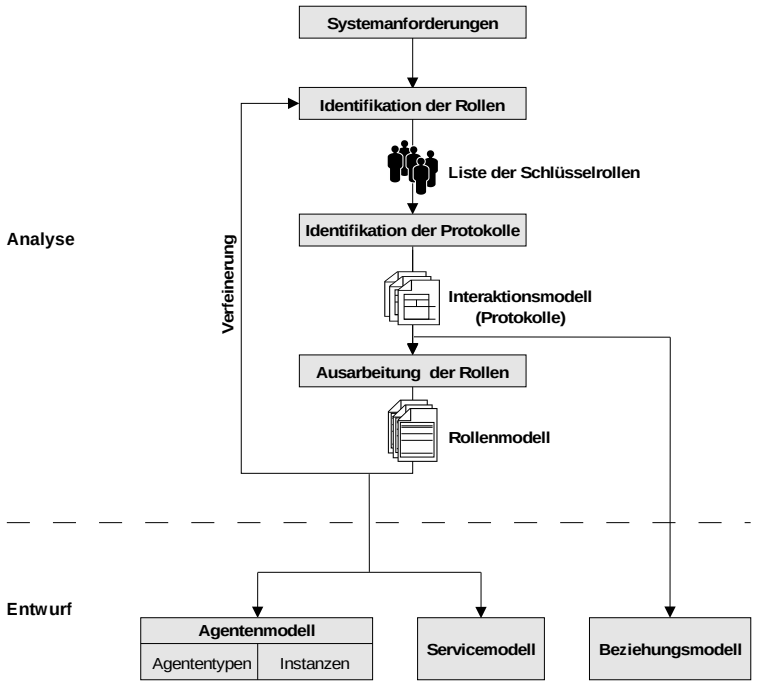


Abbildung 1.10: Der GAIA-Entwicklungsprozess (nach Weiß und Jakob, 2004, S.48)

Diese Parameter – Typisierung, Interaktionsstruktur und Instantiierung – werden durch das *Organisationsdesign* festgelegt. Wir betrachten exemplarisch den GAIA-Entwicklungsprozess (vgl. Abbildung 1.10). In der Analysephase werden im ersten Schritt aus den Systemanforderungen die relevanten Rollen extrahiert und im zweiten Schritt die Interaktionen zwischen ihnen erhoben. Diese beiden Elemente werden in der Entwurfsphase genutzt, um einerseits die Agentenmodelle – inklusive der von ihnen definierten Dienste (Servicemodell) – und andererseits die Beziehungen (Beziehungsmodell) zwischen ihnen abzuleiten.

1.5 Soziale Theorien der Agentensysteme

Setzt man, wie in der Organisationsforschung lange Zeit üblich, eine Organisation mit ihren Formalstrukturen gleich, so gerät dabei aus dem Blickfeld, dass in Organisationen noch mehr Kräfte am Werk sind, als mit solch einer funktionalen Sichtweise wahrgenommen werden können. Um dies abzubilden wurde die Kategorie der informellen Strukturen und Gruppen eingeführt und mit ihnen u.a. auf die persönlichen Befindlichkeiten der Arbeiter geschaut.

In Organisationen vermischen sich in den Akteursbeziehungen informelle und formale Organisationsanteile. Als formale Strukturen gelten im allgemeinen die offiziellen, kodifizierten Teile der Organisation. Sie werden oft gleichgesetzt mit Regeln, Verfahren oder Vorschriften. Wohingegen informelle Strukturen meist die offiziell nicht vorgesehenen oder sogar heimlichen und unerlaubten Praktiken der Organisation umschreiben.

1.5.1 Sozionik: Institutionen als Objekt sozialer Prozesse

Was durch eine Einordnung in die Kategorien formal/informell weitgehend unberücksichtigt bleibt, ist die Tatsache, dass so getrennt wird, „was in der Realität untrennbar und unentwirrbar verbunden ist“ (Friedberg, 1995, 144). Denn in Wirklichkeit existiert die Formalstruktur in Abhängigkeit von den Verhaltensweisen und Praktiken, die sie zu regeln versucht. Die formalen Strukturen erlangen nur in dem Maße Wirksamkeit, wie sie in die real stattfindenden Verhaltensweisen und Praktiken der Akteure aufgenommen werden. Formalstrukturen sind das Ergebnis von Regelsetzungen oder Verhandlungen der Organisationsmitglieder und drücken dieses in kodifizierter Form aus. Formale Vorschriften werden aber ständig von den Praktiken der Beteiligten geformt, indem jene versuchen, Zwänge zu umgehen und Situationen aus- und umzudeuten.

Soziologisch gesehen ist eine Organisation daher sowohl steuernder Kontext als auch von den Akteuren gestaltete Umwelt. Organisation sind Institutionen, d.h. soziale Regelkomplexe und als solche soziale Konstrukte.

Dieser Gedanke findet sich in ähnlicher Form auch in der VKI. Der auf Logik basierende Ansatz von Castelfranchi und Conte (1995a) befasst sich ebenfalls mit der wechselseitigen Konstitution. Ausgehend vom Individuum wird diskutiert, wie Konventionen oder Normen in der Handlungslogik des Akteurs zu verankern sind. Die duale Fragestellung, nämlich wie Normen aus den Handlungen entstehen, muss – wie die Autoren selbst feststellen – jedoch weitestgehend unbeantwortet bleiben, da die Autoren kein Theoriekonstrukt neben dem Akteur einführen (wollen), so dass sie das Konzept einer überindividuell etablierten Norm nicht im Modell verankern können.

Angesichts dieser Problematik spricht sich Castelfranchi (2000) dafür aus, in den Modellen sowohl den Agenten als auch die Systemprozesse als Systemelemente zu begreifen und deren Wechselwirkungen als Modellmechanismen.[5]

Wir können daher schließen, dass für die Modellierung einer wechselseitigen Beeinflussung soziale Strukturen ebenso wie Agenten im Modell zu repräsentieren sind. Diese Wechselseitigkeit ist uns bereits als Mikro-Makro-Dualität bekannt. Die Forderung kommt der Situation in Abbildung 1.3 gleich. Das Wechselspiel von Mikro- und Makro-Elementen muss Bestandteil des Systementwurfs sein. Dazu müssen diese beiden Elemente jeweils im Modell abgebildet werden. Wir werden auf diesen Punkt in Kapitel **??** erneut zurückkommen.

Das DFG-Schwerpunktprogramm *Sozionik* widmet sich genau der Integration dieser beiden Perspektiven sowie deren Wechselseitigkeit.

> „Eine der Aufgaben sozionischer Forschungsarbeiten besteht darin, Mikro-Makro-Zusammenhänge auf den unterschiedlichen Ebenen sozia-

[5] “[...] I claimed that only agent based social simulation joint with AI models of agents can eventually solve this problem by formally modelling and simulation *at the same time* the individual minds and behaviours, the ermegent collective action, structure or effect, and their feedback to shape minds and reproduce themselves.” (Castelfranchi, 2000, S.5)

> ler Koordination derart zu rekonstruieren, dass damit die Grundlagen für große, fehlerfreundliche und rekursiv vernetzte Multiagentensysteme gelegt werden.“ (Sozionik, 1998)

Da Agentensysteme ihrer Natur nach soziale Systeme sind, ist es naheliegend die Mikro-Makro-Wechselwirkungen aus der Perspektive der Sozialwissenschaften zu untersuchen. Der Absatz der Sozionik ist es, die dortigen Konzeptionen von Sozialität sozialwissenschaftliche Theorien für das Design von Agentensystemen aufzugreifen. „Die moderne Gesellschaft bietet [...] ein reichhaltiges Reservoir an Vorbildern für die Modellierung von Multiagentensystemen. Dabei kann die Informatik von der Adaptivität, Robustheit, Skalierbarkeit und Reflexivität sozialer Systeme lernen und ihre Bauprinzipien in leistungsfähige Technologien umsetzen.“ (Sozionik, 1998)

Es verwundert nicht, dass in der Sozialwissenschaft ein ähnlicher Perspektivenpluralismus herrscht wie in der informatischen Theorie. So denkt Gesellschaftstheorie Soziales primär von den Sozialstrukturen her, Akteurstheorie von den Akteuren und soziologische Netzwerktheorie von den Interaktionen. Prinzipiell kann die Mikro-Makro-Wechselwirkung aus einer speziellen Perspektive – Agent, Gruppe, Organisation oder Sozialität – verstanden werden. Eine solche einseitige Fokussierung ist aber bzgl. des Beschreibungsaufwandes bestenfalls unökonomisch – praktisch gesehen tendiert eine Fokussierung jedoch dazu, unvollständig zu bleiben, da die wechselseitige Konstruktion der Elemente verschiedener Perspektiven leicht aus dem Blickfeld gerät. Wir stellen somit die Wechselwirkung in den Vordergrund unserer Analysen, womit auch gleichzeitig klar ist, dass sowohl Mikro- als auch Makro-Elemente Theorieobjekte sein müssen.

Speziell die Ergebnisse in Bezug auf die konstituierenden Parameter der Mikro-Makro-Prozesse besitzen einen Bezug zur Skalierungsproblematik in Agentensystemen. Unsere Aufgabe ist es, die gefundenen Mechanismen zu qualitativen Erweiterung der MAS/VKI-Konzepte – wie Kontrolle, Verhandlung, Kooperation etc. – zu nutzen. Wenn wir soziale Strukturen im Multiagentensystem repräsentieren, dann sind wir auch in der Lage die Selbstorganisation auf Ebene des Systems zu beschreiben. Verwandte Ansätze finden sich im Bereich der Adaption von Organisationsformen bei Turner und Jennings (2001), Panzarasa und Jennings (2001), Glaser und Morignot (1997), Kirn und Gasser (1998) und Schillo (2003).

1.5.2 Reflexivität in der agentenorientierten Softwareentwicklung

Die Betrachtung des GAIA-Entwicklungsprozesses im vorangegangenen Abschnitt lässt darauf schließen, dass die vom ihm generierten Agentensysteme nicht darauf ausgelegt sind, sich an dynamische Systemanforderungen anzupassen. Bei Veränderung der Anforderungen kommt es zu einer Fortentwicklung des Systems. Design- und Implementationsphasen sind daher zyklisch verschränkt (vgl. Abbildung 1.12). Es ergibt sich ein iteratives Entwicklungsmodell, wie beispielsweise das des evolutionären Prototyping (Floyd, 1993).

Für iterative Entwicklungsmodelle ist es wichtig, dass die *konzeptionelle Lücke* zwischen der Design- und der Implementationssprache möglichst klein ist, da sich dann die Abbildbarkeit von Design und Implementation besser (d.h. korrekt und damit billig) realisieren lässt. Idealerweise sind die Modellbausteine der Designsprache bereits Bausteine (engl. *first order objects*) der Implementationssprache. Fallen die Konzepte beider Darstellungssysteme zusammen, so ist es naheliegend, nicht

mehr den Entwickler allein die Anpassungen an sich verändernde Systemanforderungen vornehmen zu lassen, sondern auch das System selbst. Die Fortentwicklung geschieht durch das System selbst (vgl. Abbildung 1.12). Diese Idee findet derzeit industriellen Anklang im Bereich des *Autonomous Computing* von IBM (Kephart und Chess, 2003).

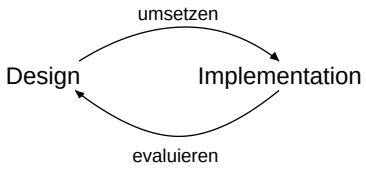


Abbildung 1.11: Iterative Entwicklung

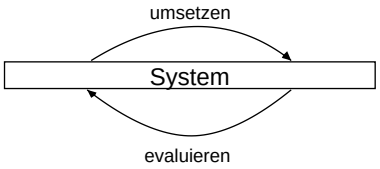


Abbildung 1.12: Selbstorganisation

Für uns ergibt sich daher im folgenden die Frage nach der Flexibilität und der Dynamik von Organisationsstrukturen. Organisationsstrukturen sind flexibel, wenn sie sich vom System selbst modifizieren lassen. Dabei stellt sich noch die Frage, wie groß der Aufwand ist, sich an veränderliche Umwelten anzupassen.

Ist von Interesse, dass nicht nur der Entwickler sie modifizieren kann, sondern dass Veränderungen auch als emergentes Produkt der Agenteninteraktionen enstehen können, dann ist es notwendig, die Strukturen *explizit* im Modell zu repräsentieren. Organisationsstrukturen, die der Veränderung der Organisation dienen, bezeichnen wir als Meta-Strukturen. Mit ihnen nimmt die Organisation auf sich selbst Bezug.

Es ist festzuhalten, dass die meisten Entwicklungsmodelle Organisationsstrukturen zwar als Analyse-, nicht aber als Implementationskategorie führen, d.h. dass in der Modellierungssprache keine Konstrukte existieren, die Organisationsstrukturen direkt auszudrücken.[6] Organisationsstrukturen sind also keine *first order objects*, sie werden stattdessen auf andere Elemente abgebildet. Beispiele für solche Entwicklungsmodelle sind GAIA (Zambonelli u. a., 2003) und MaSE (DeLoach u. a., 2001). Ansätze, bei denen Organisationsstrukturen sowohl Gegenstand der Analyse als auch der Implementation sind, finden sich bei MOISE (Hannoun u. a., 2000), Taems (Nagendra u. a., 1996), OMNI (Dignum u. a., 2004) und Tropos (Bresciani u. a., 2004).

Abbildung 1.13 illustriert die Reflexivität der agentenorientierten Softwareentwicklung, indem es im Modell der Softwareentwicklung aus Abbildung 1.9 berücksichtigt, dass das zu entwickelnde Softwaresystem eine Multiagentensystemorganisation wie in Abbildung 1.5 ist. Es zeigt sich hier die Strukturgleichheit von Entwicklung und Entwickelten.

Zusammenfassung

Die in diesem Kapitel betrachteten Konzepte der Multiagentensysteme existieren in interaktionsorientierten und in struktureller Sichtweise. Hierbei geht es insbesondere um den Aspekt der Koordinierung in Agentensystemen. Wir haben festgestellt,

[6] Schwarmsysteme sind ein Beispiel eines Ansatzes, bei dem Organisationsstrukturen weder in der Analyse- noch in der Implementationsphase berücksichtigt werden. Ein weiterer Grenzfall sind Systeme, die Organisationsstrukturen nur ad-hoc aufbauen, beispielsweise im Rahmen des Kontraktnetzprotokolls.

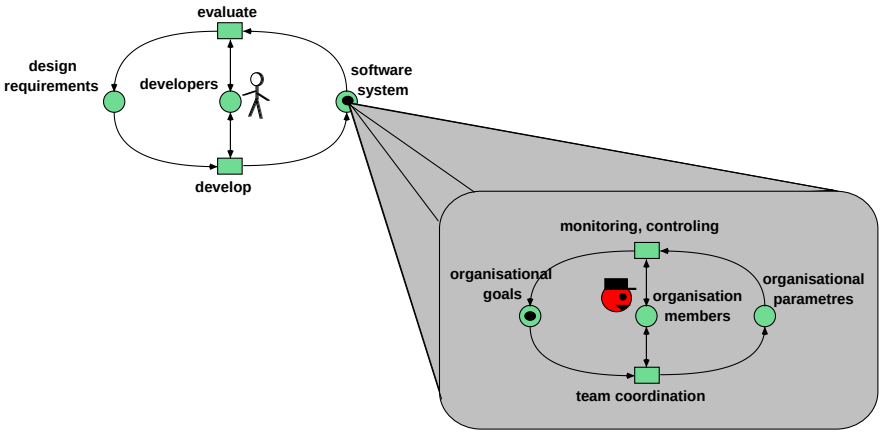


Abbildung 1.13: Reflexivität der agentenorientierten Softwareentwicklung

dass Multiagentensysteme auf verschiedenen Abstraktionsebenen betrachtet werden können. Wir können zum einen die Perspektive der Agenten und ihrer Handlungswahl einnehmen. Abstraktere Perspektiven betrachten Agentensysteme dagegen auf Ebene der Interaktion bzw. auf der Ebene des Teams. Die abstrakteste Perspektive ist wohl die, ein Agentensystem als Organisationsform zu betrachten.

Jede Perspektive liefert ihren spezifischen Beitrag zur Flexibilität der Agentensysteme. Koordination in Form von Kooperation und Konkurrenz stellt einen Mechanismus auf der Ebene kleinerer Agentengruppen dar, um durch Verhandlung und verteiltes Problemlösen Aufgaben intelligent und flexibel zu bearbeiten. Daneben haben wir mit Normen und Institutionen überindividuelle Mechanismen kennengelernt, die sich auf die Handlungswahl der Agenten auswirkt, ohne selbst direkt Bestandteil des Agenten zu sein. Offensichtlichste Institutionsform im Kontext der Multiagentensysteme sind Organisationen, die insbesondere für die agentenorientierte Softwareentwicklung als eine von den Agenten unabhängige Vorstrukturierung des Systems dienen.

Als besondere Herausforderung haben wir die Skalierungsproblematik der Verteilten Künstlichen Intelligenz kennengelernt, d.h. die Aufgabe auf Basis lokaler Programmierung effektiv und effizient global kohärente Prozesse zu erschaffen. In diesem Zusammenhang begegneten wir der Wechselseitigkeit von Handlungen und Strukturen: Zum einen bedingen die Organisationsstrukturen des Agentensystems die Handlungen der Agenten, indem sie jene rahmen und einschränken. Aber zum anderen sind es auch die Handlungen, die bestehende Strukturen verändern und sogar neue hervorbringen.

Die Sozionik studiert genau diese Wechselwirkung von Mikro- und Makro-Elementen in Multiagentensystemen, d.h. die Abhängigkeiten von Prozessen, bei denen zum einen die Handlunsgwahl des Akteurs eine unabhängige Variable ist, zum anderen aber auch die Institutionen eine Rolle spielen. Die Herausforderung liegt hierbei in der reflexiven Verschränkung beider Effekte.

Betrachten wir nun in den folgenden Abschnitten, wie diese Verschränkung von sozialen Theorien beschrieben wird und wie sich die beschriebenen Mechanismen auf Multiagentensysteme übertragen lassen.

2 Interaktion verteilter Systeme

Im folgenden wollen wir verteilte Rollen-Interaktionen beschreiben, d.h. Abläufe, bei denen die Handlungen von Akteuren ausgeführt werden, die eine Rolle wahrnehmen. Solche Interaktionen lassen sich durch Interaktionsdiagramme beschreiben, bei denen jeder Rolle eine Lebenslinie zugeordnet ist.

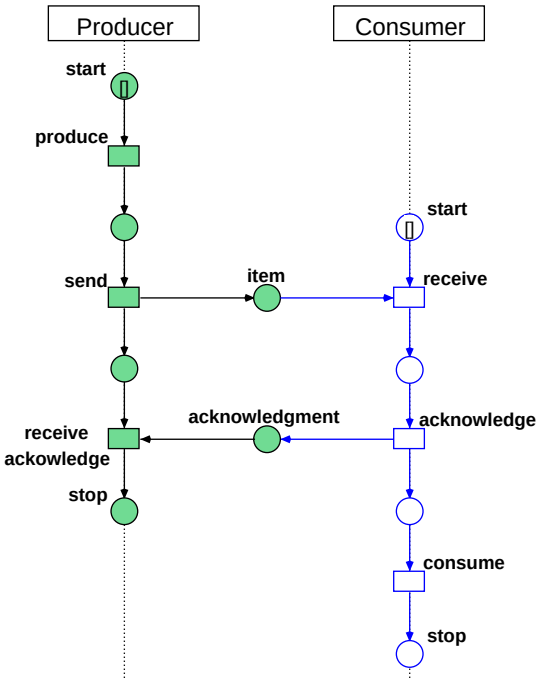


Abbildung 2.1: PC_1: Das Dienstnetz: Producer/Consumer

Um das Verhalten solcher Diagramme formal beschreiben zu können, entwickeln wir in diesem Kapitel den Formalismus der *Dienstnetze*.[1] In der Literatur existieren zahlreiche petrinetzbasierte Ansätze, die sich mit der Interaktionsperspektive in Multiagentensystemen auseinandersetzen (siehe dazu Holvoet, 1995; Chainbi u. a., 1996; Moldt und Wienberg, 1997; Billington u. a., 1998; Gois u. a., 1998; Fiorino und Tessier, 1998; Cost u. a., 1999; Hameurlain u. a., 1999; Köhler u. a., 2001; Cabac u. a., 2003). Abbildung 2.1 zeigt das dem Interaktionsdiagramm für das *Producer/Consumer* Szenario zugrundeliegende Dienstnetz. Es sind die beiden Rollen *Producer* und *Consumer* beteiligt.

Jeder Rolle des Dienstnetzes wird eine Komponente zugeordnet. In dem Beispiel stellen die ausgefüllten Netzelemente die Producer-Komponente dar. Im praktischen Kontext besitzen Dienstnetze die Eigenschaften, dass jeder Rolle R eine Interaktionslinie zugeordnet sind, so dass die R-Komponente aus einer Lebenslinie (evtl. mit AND- oder OR-Verzweigungen) besteht, deren Randstellen gerade die Nach-

[1] Dieser Abschnitt erweitert die Vorarbeiten in (Köhler und Ortmann, 2005; Köhler u. a., 2006).

richten sind. In (Cabac u. a., 2003) bilden solche R-Komponenten die Grundlage zur Generierung von Protokollen aus Dienstnetzen.

2.1 Modellierung von Interaktionen durch Dienstnetze

Im folgenden wollen wir Netze mit gefärbten Marken betrachten. Es existieren drei prominente Formalismen gefärbter Netze: die *Prädikaten/Transitionsnetze* nach Genrich und Lautenbach (1981), die *algebraischen Netze* nach Reisig (1991) und die *Coulored Petri Nets* nach Jensen (1992). (Für eine Einführung siehe Jensen und Rozenberg, 1991).Wir formalisieren Dienstnetze im folgenden als algebraische Netze, bei denen die Marken Elemente eines algebraischen Datentyps sinds. Wir nehmen dabei die Signatur nicht als direkt gegeben an, sondern generieren sie anhand einer Konzeptbeschreibung.

2.1.1 Ablaufnetze

Wir betrachten im folgenden ungewichtete Netze $N = (P, T, F)$, die keine isolierten Transitionen besitzen, d.h. $\forall t \in T : {}^\bullet t \neq \emptyset \wedge t^\bullet \neq \emptyset$. Solche Netze heißen *$T$-schlicht*.

Wir fordern, dass jeder Netzknoten auf einem Pfad zwischen den Randknoten des Netzes liegt.[2] Mit der Notation ${}^\circ N = \{n \in (P \cup T) \mid {}^\bullet n = \emptyset\}$ und $N^\circ = \{n \in (P \cup T) \mid n^\bullet = \emptyset\}$ für die Randknoten heißt dies:

$$\forall n \in (P \cup T) : \exists i \in {}^\circ N, o \in N^\circ : iF^*nF^*o \tag{2.1}$$

Man beachte, dass für T-schlichte Netze ${}^\circ N, N^\circ \subseteq P$ gelten muss.

Definition 1 *Ein* Ablaufnetz *ist ein T-schlichtes Petrinetz $N = (P, T, F)$, bei dem jeder Knoten $n \in P \cup T$ auf einem Pfad zwischen einem ${}^\circ N$ und N° liegt.*

Die Markierung m_i, in der alle Stellen aus ${}^\circ N$ einfach markiert sind, heißt kanonische Initialmarkierung, die Markierung m_f, in der nur die Stellen aus N° markiert sind, heißt kanonische Finalmarkierung:

$$m_i(p) = \begin{cases} 1, & \textit{falls } p \in {}^\circ N \\ 0, & \textit{sonst} \end{cases} \quad \textit{und} \quad m_f(p) = \begin{cases} 1, & \textit{falls } p \in N^\circ \\ 0, & \textit{sonst} \end{cases}$$

2.1.2 Konzept-Spezifikation

Algebraische Netze nutzen *abstrakte Datentypen* zur Spezifikation der Marken. Die Spezifikation eines abstrakten Datentyps besteht aus einer formalen Sprache, der Signatur, die durch eine Algebra interpretiert wird (vgl. dazu Ehrig und Mahr, 1985; Loeckx u. a., 1996). Wir betrachte im folgenden mehrsortige Signaturen.

Eine (heterogene) *Signatur* $\Sigma = (K, \Omega)$ besteht aus einer endlichen Menge an *Sorten* (engl. „kinds“) K, einer endlichen indizierten Menge $\Omega = \{\Omega_{w,k}\}_{w \in K^*, k \in K}$ von *Operatoren*. In Anlehnung an die Notation für Funktionen werden Operatoren kurz als $\omega : k_1 \times \ldots \times k_n \to k$ für $\omega \in \Omega_{k_1 \ldots k_n, k}$ notiert. Operatoren $\omega \in \Omega_{\lambda,k}$ bezeichnen Konstanten der Sorte k.

[2] Würden wir nur fordern, dass alle Knoten zwischen ${}^\circ N$ und N°, sondern zwischen Teilmengen $P_i \subset {}^\circ N$ und $P_f \subset {}^\circ N$ liegen, dann kann eine Stelle $p \in {}^\circ N \setminus P_i$ nicht zwischen P_i und P_f liegen, da ${}^\bullet p = \emptyset$ gilt. Analog liegt keine Stelle $p \in N^\circ \setminus P_f$ zwischen P_i und P_f, da $p^\bullet = \emptyset$ gilt. Also müssen wir stets den kompletten Rand betrachten.

Eine Erweiterung der mehrsortigen Signaturen stellt die *Membership Equational Logic* (MEL) (Meseguer, 1997; Bouhoula u. a., 2000) dar. Membership Equational Logic stellt eine konservative Erweiterung der *many-sorted algebra* sowie der *order-sorted algebra* dar. Zusätzlich zu den Sorten $k \in K$ werden Teilsorten $s \in S_k$ definiert, wobei mehrere Teilsorten einer Sorte zugeordnet werden. Werden keine Teilsorten definiert, so entspricht die Membership Equational Logic einer mehrsortigen Signatur. In jeder Algebra einer Membership Equational Logic sind die Träger der Teilsorten durch Teilmengen der Trägermenge der Hauptsorte definiert. Elemente der Teilsorten korrespondieren intuitiv zu den wohlgeformten Objekten der allgemeinen Hauptsorte. Eine Membership Equational Logic erlaubt es, Aussagen über das Enthaltensein eines Terms in einer Teilsorte zu treffen.

Definition 2 *Eine MEL-Signatur $\Sigma = (K, \Omega, S)$ besteht aus einer endlichen Menge an Sorten K, so dass (K, Ω) eine Signatur ist, und einer K-indizierten Menge $S = (S_k \mid k \in K)$ an paarweise disjunkten* Teilsorten*, wobei $S_k \cap K = \emptyset$ angenommen wird.*

Sei eine Signatur $\Sigma = (K, \Omega)$ und eine indizierte Variablenmenge $X = (X_k \mid k \in K)$ mit disjunkten Variablenmengen X_k gegeben.

Definition 3 *Die Menge der* Terme $\mathbb{T}^k_\Sigma(X)$ *der Sorte k über den Variablen X ist induktiv definiert:*

1. *Für alle Variablen $x_k \in X_k$ gilt $x_k \in \mathbb{T}^k_\Sigma(X)$.*
2. *Für alle Konstanten $\omega : \lambda \to k$ gilt $\omega \in \mathbb{T}^k_\Sigma(X)$.*
3. *Sei $\omega : k_1 \cdots k_n \to k$ und $t_i \in \mathbb{T}^{k_i}_\Sigma(X)$ für alle $1 \leq i \leq n$, dann gilt $\omega(t_1, \ldots, t_n) \in \mathbb{T}^k_\Sigma(X)$.*

Dann bezeichnet $\mathbb{T}_\Sigma(X) := \bigcup_{k \in K} \mathbb{T}^k_\Sigma(X)$ die Menge aller Terme mit den Variablen X. Die Menge $\mathbb{T}_\Sigma := \bigcup_{k \in K} \mathbb{T}^k_\Sigma$ mit $\mathbb{T}^k_\Sigma := \mathbb{T}^k_\Sigma(\emptyset)$ enthält alle Grundterme.

Sei $t \in \mathbb{T}^k_\Sigma(X)$ ein Term. Die Menge der verwendeten Variablen $\textsc{Var}(t)$ von t ergibt sich mit der induktiven Termstruktur und den Definitionen $\textsc{Var}(x) := \{x\}$ für $x \in X_k$ und $\textsc{Var}(\omega(t_1, \ldots, t_n)) := \bigcup_{i=1}^n \textsc{Var}(t_i)$.

Seien $t, w \in \mathbb{T}_\Sigma(X)$ Terme, dann wird durch $t[w/x]$ die Substitution im Term t von der Variable x durch den Term w bezeichnet. Die simultane Substitution wird mit $t[w_1, \ldots, w_n/x_1, \ldots, x_n]$ bezeichnet.

Die atomaren Formeln, die mit den Termen der Signatur gebildet werden, sind entweder Aussagen über das Enthaltensein eines Terms in einer Teilsorte (engl. „membership“) oder die Gleichheit zweier Terme.

Definition 4 *Jede* MEL-Formel ϕ *ist von der Form:*

1. *$t \in s$ für $s \in S_k$ und $t \in \mathbb{T}^k_\Sigma(X)$.*
2. *$t_1 = t_2$ für $t_1, t_2 \in \mathbb{T}^k_\Sigma(X)$.*
3. *$(\phi_1 \wedge \ldots \wedge \phi_n)$ für Formeln ϕ_i.*

Ein MEL-Axiom *hat die Form (für eine MEL-Formel ϕ):*

1. *$\forall X : \phi \Longrightarrow (t \in s)$.*

2. $\forall X : \phi \Longrightarrow (t_1 = t_2)$.

Eine Datentyp-Spezifikation $\mathcal{S} = (\Sigma, X, E)$ *besteht aus einer Signatur* $\Sigma = (K, \Omega, S)$*, einer disjunkten Familie von Variablen* $X = (X_k \mid k \in K)$ *und einer Familie* $E = (E_k \mid k \in K)$ *von Axiomen.*

Die Menge aller MEL-Formeln wird mit $MEL_{\mathcal{S}}$ *bezeichnet.*

Auch wenn die Definition für die Axiome $\forall X : \phi \Longrightarrow \psi$ nur die Konjunktion für ϕ erlaubt, kann die Disjunktion $\forall X : (\phi_1 \vee \phi_2) \Longrightarrow \psi$ doch mit der logischen Äquivalenz $(A \vee B) \Longrightarrow C \equiv (A \Longrightarrow C) \wedge (B \Longrightarrow C)$ durch zwei Axiome formuliert werden.

Man beachte, dass nur Terme der Sorten $k \in K$, nicht aber der Teilsorten s definiert sind. Mit den Teilsortenformeln können wir aber eine praktische Kurznotation vereinbaren: Im folgenden verwenden wir die Notation $t \in \mathbb{T}^s_\Sigma(X)$ für eien Teilsorte $s \in S_k$, wenn $t \in \mathbb{T}^k_\Sigma(X)$ und $t \in s$ gilt.

Wenn eine Datentyp-Spezifikation keine Teilsorten definiert, so entspricht die Spezifikation in Membership Equational Logic einer mehrsortigen Signatur. Das klassische Beispiel einer solchen Sigantur ist die Sorte NAT der natürlichen Zahlen, die durch die Operatoren $0 \in \Omega_{\lambda,\mathrm{NAT}}$ und $suc \in \Omega_{\mathrm{NAT},\mathrm{NAT}}$ beschrieben wird. Daraus leitet sich die Addition als Gleichungsspezifikation ab. In der MAUDE-Syntax (Maude, 1999) wird dies folgendermaßen notiert:

```
fmod NAT is
  sort Nat .
  op 0 : -> Nat .
  op suc : Nat -> Nat .
  op _+_: Nat Nat -> Nat .

  vars M N : Nat .
  eq 0 + N = N .
  eq suc(M) + N = suc(M + N) .
endfm
```

In diesem Beispiel können die Gleichungen als Ersetzungsregeln von links nach rechts angewendet werden. Da Termination und Konfluenz gilt, erzeugt diese Reduktion eine Normalform der Gestalt $suc^n(0)$.

Eine Algebra zu einer Signatur besteht aus einer indizierten Familie von Trägermengen zusammen mit einer Familie von Funktionen.

Definition 5 *Eine* Σ-Algebra $A = (\llbracket K \rrbracket_A, \llbracket \Omega \rrbracket_A, \llbracket S \rrbracket_A)$ *zu der Signatur* $\Sigma = (K, \Omega, S)$ *besteht aus*

- *einer indizierten Familie disjunkter Trägermengen* $\llbracket K \rrbracket_A = (\llbracket k \rrbracket_A \mid k \in K)$*,*
- *einer indizierten Familie von Funktionen*

$$\llbracket \Omega \rrbracket_A = \big(\llbracket \omega \rrbracket_A : \llbracket k_1 \rrbracket_A \times \cdots \times \llbracket k_n \rrbracket_A \to \llbracket k \rrbracket_A \mid \omega \in \Omega_{k_1 \cdots k_n, k}\big)$$

- *einer indizierten Familie disjunkter Submengen* $\llbracket S \rrbracket_A = (\llbracket S_k \rrbracket_A \mid k \in K)$*, so dass für alle* $\llbracket s \rrbracket_A \in \llbracket S_k \rrbracket_A$ *auch* $\llbracket s \rrbracket_A \subseteq \llbracket k \rrbracket_A$ *gilt.*

Sei eine Spezifikation $\mathcal{S} = (\Sigma, X, E)$ gegeben. Eine *Variablenbelegung* $\alpha : X \to \llbracket K \rrbracket_A$ ist eine Familie $(\alpha_k \mid k \in K)$ von Abbildungen $\alpha_k : X_k \to \llbracket k \rrbracket_A$. Jede Variablenbelegung α induziert die Auswertungsabbildung $\overline{\alpha} : \mathbb{T}_\Sigma(X) \to \llbracket K \rrbracket_A$ auf Termen. Die *Auswertung* $\overline{\alpha}(t)$ eines Terms $t \in \mathbb{T}^k_\Sigma(X)$ in der Algebra A ist induktiv definiert:

1. $\overline{\alpha}(x_k) = \alpha(x_k)$ für $x_k \in X_k$.

2. $\overline{\alpha}(\sigma(t_1, \ldots, t_n)) = [\![\sigma]\!]_A(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_n))$ für $\sigma(t_1, \ldots, t_n) \in \mathbb{T}^k_\Sigma(X)$.

Definition 6 *Die Gültigkeit einer MEL-Formel ϕ unter einer Belegung α ist definiert durch:*

1. $A, \alpha \models (t \in s)$, *gdw.* $\alpha(t) \in [\![s]\!]_A$.

2. $A, \alpha \models (t_1 = t_2)$, *gdw.* $\alpha(t_1) = \alpha(t_2)$.

3. $A, \alpha \models (\phi_1 \wedge \ldots \wedge \phi_n)$, *gdw.* $A, \alpha \models \phi_i$ *für alle* $1 \leq i \leq n$.

Ein Axiom $\forall X : \phi \Longrightarrow \phi'$ ist in der Algebra A gültig (notiert als $A \models \forall X : \phi \Longrightarrow \phi'$), gdw. ϕ' in allen Belegungen α gültig ist, für die auch ϕ gültig ist.
Eine Σ-Algebra A, für die alle Axiome der Spezifikation $\mathcal{S} = (\Sigma, X, E)$ gültig sind, heißt $\mathcal{S}$-Theorie.

Eine Signatur Σ generiert die Kategorie Σ-*Alg*, die als Objekte Σ-Algebren und als Morphismen Σ-Algebra-Morphismen besitzt. Die allgemeinste Algebra ist in die *Termalgebra* $T(\Sigma)$, die jeden Operator ω in Σ durch sich selbst interpretiert, d.h. für alle $t_i \in \mathbb{T}^{k_i}_\Sigma$, $1 \leq i \leq n$ gilt:

$$[\![\omega]\!]_{T(\Sigma)}(t_1, \ldots, t_n) = \omega(t_1, \ldots, t_n)$$

Die Termalgebra ist das initiale Objekt der Kategorie Σ-*Alg*, d.h zu jeder Σ-Algebra A existiert ein eindeutig bestimmter Homomorphismus $h_A : T(\Sigma) \to A$. Die Termalgebra ist zudem noch minimal und enthält keine trivialen Gleichungen.

Analog generiert jede Spezifikation $\mathcal{S} = (\Sigma, X, E)$ die Kategorie $\mathcal{S}$-*Alg* aller $\mathcal{S}$-Algebren. Die Kategorie aller $\mathcal{S}$-Algebren besitzt ebenfalls ein initiales Objekt, nämlich die Kongruenzalgebra.Die *Kongruenzalgebra* $\mathbb{T}_{\Sigma,E} := \mathbb{T}_\Sigma / \equiv_E$ der Termalgebra $\mathbb{T}_\Sigma$ ist das initiale Objekt der Kategorie $\mathcal{S}$-*Alg*. Die Mengen $\mathbb{T}_{\Sigma,E}$ und $\mathbb{T}_{\Sigma,E}(X)$ bezeichnen die Algebra der Kongruenzklassen-Terme bezüglich der Gleichungsmenge E.

2.1.3 Konzeptbeschreibung

Teilsorten werden eingesetzt, um wertabhängigige Typisierungen aussprechen zu können. Betrachten wir dazu das Beispiel eines Graphen (V, E) mit Knotenmenge V und Kantenmenge E.

Eine Kantenwort ist ein Element $w \in E^*$, also ein Element des freien Monoids $(E^*, \cdot, \epsilon)$. Nun ist aber nicht jedes Kantenwort auch ein Pfad im Graph. Ein Pfad w des Graphen zeichnet sich dadurch aus, dass für jede Zerlegung $w = w_1 \cdot w_2$ gilt, dass der Endknoten von w_1 mit dem Anfangsknoten von w_2 identisch ist. Dies ist aber eine Bedingung, die wir nicht mehr auf der Ebene der Sorte der Worte direkt festmachen können, da die Eigenschaft, ob $w_1 \cdot w_2$ ein Pfad ist, nicht nur von $\cdot$, sondern auch an den Attributen von w_1 und w_2 abhängt. Diese wertabhängige Typisierung der Teilsorte Path lässt sich leicht in Membership Equational Logic formulieren. Sei e eine Kantenvariable und p eine Pfadvariable, dann fomulieren wir das Axiom:

$$\forall e, p : \text{target}(e) = \text{source}(p) \Longrightarrow (e \cdot p) \in \mathsf{Path}$$

Die Sorte `Seq` beschreibt alle potentiellen Pfade, die syntaktisch durch den Operator `;` bildbar sind. Ein echter Pfad des Graphen wird durch die Sorte `Path` beschrieben. Jede Kante ist ein echter Pfad, was durch die Notation `subsorts Edge < Path` ausgedrückt wird. Das obige Axiom lautet in MAUDE-Syntax:

```
cmb (E ; P) : Path if target(E) == source(P) .
```

Hierbei steht *cmb* für *conditonal membership*. Die gesamte Spezifikation ergibt sich wie folgt:

```
sorts Edge Node Path Seq .
subsorts Edge < Path < Seq .

ops source target : Edge -> Node .
op _;_ Seq Seq -> Seq [assoc] .
ops source target : Path -> Node .
op length : Path -> Nat .

var E : Edge .  var P : Path .

cmb (E ; P) : Path if target(E) == source (P) .

eq source(E ; P) = source(E) .
eq target(E ; P) = target(P) .

eq length(E) = suc(0) .
eq length(E ; P) = suc(0) +  length(P) .
```

In diesem Beispiel werden Subsorten verwendet, um – beispielsweise für den Operator `length` – Pfade nicht erst auf der Termebene, sondern schon auf Ebene des Typsystems auszuzeichnen. Beispielsweise erwartet der Operator `length` eine Argument vom Typ `Path`. In der Gleichung `eq length(E ; P) = suc(0) + length(P)` ist `(E ; P)` zunächst vom Typ `Seq`. Es ist also notwendig, zunächst `(E ; P) : Path` herzuleiten, bevor die Gleichung angwendet werden kann.

Die Membership Equational Logic ist reich genug, um bereits Typkonstruktionen wie Schnitt oder Vereinigung darzustellen. Wir vergleichen im folgenden das Verhältnis der memberships zu der Subsumptionsbeziehung der Beschreibungslogik.

2.1.4 Beschreibungslogik

Beschreibungslogik bildet Formeln über elementaren Konzepten und Relationen zwischen ihnen (die als *Rollen* bezeichnet werden – nicht zu Verwechseln mit den Rollen der nachfolgenden Dienstnetze), die durch Operationen verknüpft werden können. Die Subsumption $\sqsubseteq$ ist das alleinige Prädikat der Logik.

Betrachten wir ein Beispiel: Seien Woman und Man atomare Konzepte und hasChild die Eltern/Kind-Relation. Eine mögliche darauf aufbauende Terminologie (die T-Box) beschreibt abgeleitete Konzepte und Relationen:

$$\begin{aligned}
\mathsf{Person} &\equiv \mathsf{Woman} \sqcup \mathsf{Man} \\
\mathsf{Mother} &\equiv \mathsf{Woman} \sqcap \exists \mathsf{hasChild}.\mathsf{Person} \\
\mathsf{Father} &\equiv \mathsf{Man} \sqcap \exists \mathsf{hasChild}.\mathsf{Person} \\
\mathsf{Parent} &\equiv \mathsf{Mother} \sqcup \mathsf{Father} \\
\mathsf{Grandmother} &\equiv \mathsf{Woman} \sqcap \exists \mathsf{hasChild}.\mathsf{Parent}
\end{aligned}$$

Zyklischen Definitionen sind hierbei nicht erlaubt. Die Subsumption bezeichnet das Enthaltensein eines Konzeptes in einem anderen. Für unser Beispiel gilt unter anderem, dass jede Mutter eine Frau ist und jede Frau eine Person:

$$\mathsf{Mother} \sqsubseteq \mathsf{Woman} \sqsubseteq \mathsf{Person}$$

Daneben speichert eine Wissensdatenbank (die A-Box) Fakten ab:

$$\mathsf{Woman}(eva)$$
$$\mathsf{Man}(adam)$$
$$\mathsf{hasChild}(eva, abel), \mathsf{hasChild}(eva, kain)$$

Definition 7 *Seien N_C und N_R paarweise disjunkte, abzählbar unendliche Mengen von* Konzept-*, respektive* Rollennamen. *Dann ist die* Konzeptbeschreibungen *der Beschreibungslogik $\mathcal{ALC}$ folgendermaßen definiert:*

1. *Jeder atomare Ausdruck A ist eine $\mathcal{ALC}$-Konzeptbeschreibung.*
2. *$\top$ und $\bot$ sind $\mathcal{ALC}$-Konzeptbeschreibungen.*
3. *Sind C und D jeweils $\mathcal{ALC}$-Konzeptbeschreibungen und R eine Rolle, so sind auch $\neg C$, $(C \sqcup D)$, $(C \sqcap D)$, $\exists R.C$ und $\forall R.C$ auch $\mathcal{ALC}$-Konzeptbeschreibungen.*

Man erkennt, dass die Konzepte unäre und die Rollen binäre Relationen sind, die in der Beschreibungslogik allerdings in einer variablenfreien Notation verwendet werden:

$$C \sqcap D \text{ steht für } \forall x : C(x) \wedge D(x)$$

Daher werden Konzepte durch Teilmengen eines Universums und Rollen als binäre Relationen definiert.

Definition 8 *Eine* Interpretation $(\Delta, \cdot^{\mathcal{I}})$ *besteht aus einer nicht-leeren Grundmenge Δ und einer Interpretationsfunktion $\cdot^{\mathcal{I}}$:*

- *Jedem Konzeptbezeichner $A \in N_C$ wird eine Menge $A^{\mathcal{I}} \subseteq \Delta$ zugewiesen.*
- *Jedem Rollenbezeichner $R \in N_R$ wird eine binäre Relation $R^{\mathcal{I}} \subseteq \Delta \times \Delta$ zugewiesen.*

Eine Interpretation erweitert sich induktiv auf alle Konzeptbeschreibungen:

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta \\
\bot^{\mathcal{I}} &= \emptyset \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \Delta \setminus C^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \{x \in \Delta \mid \exists y : (x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} &= \{x \in \Delta \mid \forall y : (x, y) \in R^{\mathcal{I}} \Longrightarrow y \in C^{\mathcal{I}}\}
\end{aligned}$$

Damit ergibt sich die Gültigkeit folgendermaßen:

- *Ein Konzept C wird bzgl. des Modells $\mathcal{I}$ von D* subsumiert *(notiert $C \sqsubseteq D$), falls $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ gilt.*

- *Die Konzepte C und D sind bzgl. des Modells $\mathcal{I}$* äquivalent *(notiert $C \equiv D$), falls $C^{\mathcal{I}} = D^{\mathcal{I}}$ gilt.*

- *Ein Konzept heißt* erfüllbar *bzgl. des Modells $\mathcal{I}$, falls $C^{\mathcal{I}}$ nicht leer ist.*

Subsumption und Erfüllbarkeit stehen in enger Verbindung wie das folgende Theorem zeigt (Baader u. a., 2003, Theorem 2.13).

Theorem 1 *Seien C und D Konzepte, dann gilt:*

- *C wird von D subsumiert, gdw. $C \sqcap \neg D$ unerfüllbar ist.*

- *C und D sind äquivalent, gdw. $C \sqcap \neg D$ und $\neg C \sqcap D$ unerfüllbar sind.*

- *C und D sind disjunkt, gdw. $C \sqcap D$ unerfüllbar ist.*

Schmidt-Schauß und Smolka (1991) zeigten die Verbindungen der Beschreibungslogik zum Erfüllbarkeitsproblem der multimodalen Logik K, dessen Erfüllbarkeitsproblem als PSPACE-hart bekannt ist.

Theorem 2 *Das Erfüllbarkeitsproblem für $\mathcal{ALC}$ ist PSPACE-hart.*

Wir betrachten nun eine teilweise Einbettung der Beschreibungslogik in Membership Equational Logic. Wir können die elementaren Konzepte direkt als Teilsorten übernehmen. Die Rollen R betrachten wir als Prädikate $R(x, y)$, d.h. als Abbildungen in die Sorte Bool. Betrachten wir die Formelterme als Sortenbezeichner, dann können wir einige Formel direkt in Membership Equational Logic einbetten. Die Übersetzung $\phi(C)$ des Konzepts C ist folgendermaßen definiert.

$$\begin{array}{rcrcl}
\phi(C \sqcap D) & := & \forall x : (x \in C \wedge x \in D) & \Longrightarrow & x \in (C \sqcap D) \\
\phi(C \sqcup D) & := & \forall x : (x \in C \vee x \in D) & \Longrightarrow & x \in (C \sqcup D) \\
\phi(\top) & := & \forall x : \textsc{True} & \Longrightarrow & x \in \top \\
\phi(\bot) & := & \forall x : x \in \bot & \Longrightarrow & \textsc{False} \\
\phi(\exists R.C) & := & \forall x, y : (R(x, y) \wedge y \in C) & \Longrightarrow & x \in (\exists R.C)
\end{array}$$

Jede Subsortenbeziehung $x \in C \Longrightarrow x \in D$ spiegelt dabei die Subsumptionsbeziehung $C \sqsubseteq D$ der Konzeptdefinition wieder.

Die Konzeptebeschreibungen $(\neg C)$ und $\forall R.C$ sind dagegen nicht so ohne weiteres darstellbar, da man in Membership Equational Logic keine negierten Eigenschaften ausdrücken kann, so dass die folgende Darstellung der Negation sowie der Allquantifizierung keine Axiome ergeben:

$$\begin{array}{rcrcl}
\phi(\neg C) & := & \forall x : (\neg x \in C) & \Longrightarrow & x \in (\neg C) \\
\phi(\forall R.C) & := & \forall x, y : (\neg R(x, y) \vee y \in C) & \Longrightarrow & x \in (\forall R.C)
\end{array}$$

Dass die Negation nicht ausdrückbar ist, ergibt sich auch aus der Tatsache, dass Membership Equational Logic äquivalent zur Hornklausellogik mit Gleichheit ist und in Hornlogik Negation auch gesondert zu behandeln ist. Dies ist die in Prolog bekannte *negation as failure*-Semantik.

2.1.5 Rollenstrukturen

Ein Organisationsschema ist die Grundstruktur, innerhalb deren die Aktivitäten einer Organisation stattfinden. Die eigentlichen Aktivitäten werden durch Dienstnetze beschrieben. Dienstnetze sind Petrinetze, deren Transitionen Rollen zugeschrieben sind.

Rollen sind zunächst einmal nur Namen, deren Bedeutung sich erst später durch ihre Verknüpfung mit den Dienstnetzen ergeben. Sei *Rol* eine Menge von Rollen. Wir gehen im folgenden davon aus, dass die Menge *Rol* eine spezielle Rolle r_* enthält, die wir nur für formale Konstruktionen benötigen. Wir benötigen diese Rolle, da jeder Transition eines Dienstnezes eine Rolle zugewiesen ist und bei Transformation auf Dienstnetzen unter Umständen neue Transitionen hinzugefügt werden. Die Rolle ist dann aber beliebig und wird dann zu r_* gewählt. In den eigentlichen Dienstnetzen wird r_* nicht vorkommen.

Nichtleere Teilmengen $R \subseteq \mathit{Rol}$ werden als *Rollenprofile* (kurz: Profile) bezeichnet. Die einelementigen Profile $\{r\}$ mit $r \in \mathit{Rol}$ werden mit der Rolle r identifiziert.

Definition 9 *Sei Rol eine Menge von Rollen, die die spezielle Rolle r_* enthält Die* Rollenstruktur *über Rol ist die durch Mengeninklusion partiell geordnete Menge* $\mathcal{R} := 2^{\mathit{Rol}} \setminus \{\emptyset\}$.

Rollenprofile können durch Vereinigung aggregiert werden. Sie sind durch Mengeninklusion partiell geordnet: Gilt $R_1 \subseteq R_2$ für zwei Rollenprofile $R_1, R_2 \in 2^{\mathit{Rol}}$, so ist R_2 umfassender als R_1, bzw. R_1 ist spezialisierter als R_2.

2.1.6 Dienstnetze als gefärbte Petrinetze

Dienstnetze sind algebraische Ablaufnetze. Sei N ein Ablaufnetz. Das Typsystem – die *Farbspezifikation* – wird durch die Datentyp-Spezifikation $\mathcal{S} = (\Sigma, X, E)$, bestehend aus einer Signatur $\Sigma = (K, \Omega, S)$, einer disjunkten Familie von Variablen $X = (X_k \mid k \in K)$ und einer Familie $E = (E_k \mid k \in K)$ von Axiomen, spezifiziert. Indem das Typsystem sowohl die Syntax der Datenobjekte als auch die Subtypisierungsbeziehungen formalisiert, nimmt es in Dienstnetzen die Aufgabe einer Agentenkommunikationssprache wahr.

Die Farbe einer Marke ist in algebraischen Petrinetzen ein Term einer Teilsorte s. Jede Farbsorte besitzt ihre eigenen Operatoren und Gleichungen in $\mathcal{S}$. Sei $A = (\llbracket K \rrbracket_A, \llbracket \Omega \rrbracket_A, \llbracket S \rrbracket_A)$ eine Theorie zur Datentypspezifikation $\mathcal{S}$.

Jede Stelle eines Dienstnetzes ist durch die Abbildung d typisiert, d.h. alle Marken auf p sind vom Typ $d(p)$. Die Kanten algebraischer Netze werden mit Termen $W(f)$ beschriftet, die Variablen enthalten können. Der Ausdruck ist dabei von der Sorte $d(p)$, d.h. $W(f) \in \mathbb{T}^{d(p)}_{\Sigma,E}(X)$. Dabei definieren wir $d(x, y) = d(x)$, falls $(x, y) \in P \times T$ und $d(x, y) = d(y)$, falls $(x, y) \in T \times P$.

Eine *Bindung* konkreter Ausdrücke an die Variablen ergibt einen speziellen Schaltmodus. Auf diese Weise beschreibt eine Transition eine Vielzahl an möglichen konkreten Transitionen. Die Transitionen sind mit Schaltptädikaten $G(t)$ versehen. Bindungen können durch Schaltprädikate weiter eingeschränkt werden: Ein Schalten ist unter einer Bindung nur dann möglich, wenn bezüglich der Bindung genügend Marken im Vorbereich vorhanden sind und die Bindung das Prädikat erfüllt.

Eine Markierung $M : P \to MS(\llbracket K \rrbracket_A)$ des Netzes ist eine Abbildung, die jeder Stelle einen Wert $M(p) \in MS(\llbracket d(p) \rrbracket_A)$ in der Algebra A zuweist, d.h. eine Multimenge über $\llbracket d(p) \rrbracket_A$.

Jeder Markierung M von D ordnen wir eine Projektionsmarkierung $\pi(M)$ zu, indem wir die Farbe der Marke vergessen und nur noch die Anzahl der Marken zählen:

$$\pi(M)(p) = |M(p)| \tag{2.2}$$

Wir verwenden π im folgenden auch in der Erweiterung auf Markierungsmengen.

Als Besonderheit der Dienstnetze wird jeder Transition t die Rolle $r(t) \in \mathit{Rol}$ zugewiesen. Dies bedeutet, dass zur Ausführung eines Tasks ein Agent, der diese Rolle inne hat, notwendig ist. Wir erweitern die Abbildung r auf Transitionsmengen, indem wir $R(T') = \{r(t) \mid t \in T'\}$ für jede Menge $T' \subseteq T$ definieren. Die Menge der an einer Interaktion N beteiligten Rollen ist $R(D) := R(N) := R(T_N)$.

Wir fordern, dass in Dienstnetzen die Rollen konfliktfrei verbunden sind, d.h. Vor- und Nachbereich einer Stelle enthalten jeweils Transitionen mit gleicher Rollenzuweisung:

$$\forall p \in P : |R({}^\bullet p)|, |R(p^\bullet)| \leq 1 \tag{2.3}$$

Eine Stelle p heißt *Kommunikationskanal*, wenn sie Transitionen unterschiedlicher Rollen verbindet, d.h. wenn $R({}^\bullet p) \neq R(p^\bullet)$ gilt. Die Menge der Kommunikationskanäle zum Paar $(r_1, r_2) \in \mathit{Rol}^2$ ist:

$$P_{KK}(r_1, r_2) := \{p \mid R({}^\bullet p) = \{r_1\} \wedge R(p^\bullet) = \{r_2\}\} \tag{2.4}$$

Die Menge der Kommunikationskanäle ist dann:

$$P_{KK}(D) := \bigcup_{r_1, r_2 \in R(D), r_1 \neq r_2} P_{KK}(r_1, r_2) \tag{2.5}$$

Also gilt $p \in P_{KK}(D)$, wenn $R({}^\bullet p) \neq R(p^\bullet)$ und $|R({}^\bullet p)| = |R(p^\bullet)| = 1$ gilt.

Für Dienstnetze wird gefordert, dass die Rollen aus $R \subseteq R(D)$ mit dem Komplements $R(D) \setminus R$ durch einen Kommunikationskanal verbunden sind:

$$\begin{aligned} \forall R \in 2^{R(D)} \setminus \{\emptyset\} : \exists r_1 \in R : \exists r_2 \in R(D) \setminus R : \\ P_{KK}(r_1, r_2) \cup P_{KK}(r_2, r_1) \neq \emptyset \end{aligned} \tag{2.6}$$

Definition 10 *Sei Rol eine Rollenmenge. Ein* Dienstnetz

$$D = (N, A, d, r, W, G, M_0)$$

besteht aus den folgenden Komponenten:

1. *A ist eine Theorie zur Datentypspezifikation $\mathcal{S}$.*
2. *$N = (P, T, F)$ ist ein Ablaufnetz.*
3. *$d : P \to \bigcup_{k \in K} S_k$ ist die Stellentypisierung.*
4. *$r : T \to \mathit{Rol}$ ist die Rollenzuweisung, die (2.3) und (2.6) erfüllt.*
5. *$W : F \to \mathbb{T}_{\Sigma,E}(X)$ ist die Kanteninschrift.*
6. *$G : T \to \mathit{MEL}_{\mathcal{S}}$ ist die Aktivierungsbedingung.*
7. *M_0 ist eine Initialmarkierung mit $\pi(M_0) = m_0$, wobei m_0 die kanonische Initialmarkierung von N ist.*

Sei D ein Dienstnetz. Dann ist $\pi(D) := N$ das D zugrundliegende Ablaufnetz.

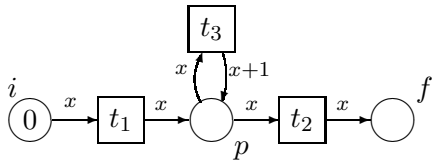


Abbildung 2.2: Ein unbeschränktes algebraisches Netz

Die Schaltregel ergibt sich mit Hilfe von Variablenbindungen. Jede *Bindung* $\alpha : X \to [\![K]\!]_A$ mit $A = ([\![K]\!]_A, [\![\Omega]\!]_A, [\![S]\!]_A)$ der Variablen, die das Aktivierungsprädikat $G(t)$ der Transition t erfüllt, erzeugt einen *Schaltmodus*.

Definition 11 *Eine Transition t unter der Variablenzuweisung $\alpha : X \to [\![K]\!]_A$ heißt genau dann in M* aktiviert*, wenn die Schaltbedingung erfüllt ist: $A, \alpha \models G(t)$ und für alle $p \in P$ auch $M(p) \geq \overline{\alpha}(W(p,t))$ gilt.*

Das Schalten der Transition t unter der Belegung α wird als $M \xrightarrow{t,\alpha} M'$ notiert, wobei sich die Nachfolgemarkierung *für jede Stelle $p \in P$ wie folgt ergibt:*

$$M'(p) = M(p) - \overline{\alpha}(W(p,t)) + \overline{\alpha}(W(t,p))$$

Aus der Schaltregel ergeben sich die Definitionen für Schaltfolgen, erreichbare Markierungen usw. analog zu einfachen Netzen. Schaltfolgen w sind Wörter über $(T \times (X \to [\![K]\!]_A))$, d.h. $w = (t_1, \alpha_1) \cdots (t_n, \alpha_n)$. Ein Dienstnetz heißt *beschränkt*, wenn die Menge der erreichbaren Markierungen endlich ist. Man beachte, dass die Anzahl der Marken auf jeder Stelle in der Anzahl beschränkt sein kann, ohne dass das Netz selbst beschränkt sein muss, wie das Netz in Abbildung 2.3 zeigt.

Jeder Schaltvorgang eines algebraischen Netzes ist auch in den Projektionen möglich. Dies halten wir im folgenden Projektionslemma fest.

Theorem 3 *Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz, dann gilt für die Projektionen:*

$$M \xrightarrow[D]{t,\alpha} M' \Longrightarrow \pi(M) \xrightarrow[\pi(D)]{t} \pi(M')$$

und

$$\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$$

Beweis: Sei $M \xrightarrow[D]{t,\alpha}$. Dies ist nach Definition erfüllt, wenn $G(t)$ gilt und wenn für alle $p \in P$ gilt:

$$\begin{array}{ll} & M(p) \geq \overline{\alpha}(W(p,t)) \\ \iff & \forall c \in [\![d(p)]\!]_A : M(p)(c) \geq \overline{\alpha}(W(p,t))(c) \\ \Longrightarrow & \sum_{c \in [\![d(p)]\!]_A} M(p)(c) \geq \sum_{c \in [\![d(p)]\!]_A} \overline{\alpha}(W(p,t))(c) \\ \iff & |M(p)| \geq |\overline{\alpha}(W(p,t))| \\ \iff & \pi(M)(p) \geq F(p,t) \end{array}$$

Also ist t auch in N aktiviert. Bei der Umformung haben wir $\pi(M(p)) = \pi(M)(p)$ und die Eigenschaft $|\overline{\alpha}(W(p,t))| = F(p,t)$ ausgenutzt. Für die Nachfolgemarkierung gilt:

$$\begin{array}{ll} & M'(p) = M(p) - \overline{\alpha}(W(p,t)) + \overline{\alpha}(W(t,p)) \\ \iff & \forall c \in [\![d(p)]\!]_A : M'(p)(c) = M(p)(c) - \overline{\alpha}(W(p,t))(c) + \overline{\alpha}(W(t,p))(c) \\ \Longrightarrow & \sum_{c \in [\![d(p)]\!]_A} M'(p)(c) = \sum_{c \in [\![d(p)]\!]_A} M(p)(c) - \overline{\alpha}(W(p,t))(c) + \overline{\alpha}(W(t,p))(c) \\ \iff & |M'(p)| = |M(p)| - F(p,t) + F(t,p) \\ \iff & \pi(M')(p) = \pi(M)(p) - F(p,t) + F(t,p) \end{array}$$

Also ist $\pi(M')$ die Nachfolgemarkierung von $\pi(M)$.

Die zweite Aussage folgt induktiv aus der ersten. q.e.d.

Aus der Eigenschaft $\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$ können wir leider nicht schließen, dass aus der Beschränktheit von $\pi(D)$ auch die Beschränktheit von D folgt, denn aus der Beschränktheit von $\pi(D)$ folgt die Endlichkeit von $RS(\pi(D), \pi(M_0))$ und aufgrund der Inklusion auch die Endlichkeit der Projektion $\pi(RS(D, M_0))$, jedoch impliziert letzteres nicht die Endlichkeit von $RS(D, M_0)$ selbst. So ist das Netz in Abbildung 2.2 unbeschränkt, da die Stelle p zwar maximal eine Marke enthält, so dass p in $\pi(D)$ beschränkt ist, die Marke aber eine beliebig große Zahl als Wert haben kann.

Die Umkehrung gilt auch nicht, denn Abbildung 2.3 ist ein beschränktes Dienstnetz D, dessen Projektion $\pi(D)$ unbeschränkt ist.

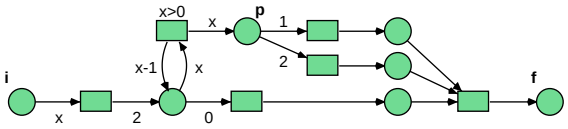


Abbildung 2.3: Ein beschränktes Dienstnetz

Theorem 4 *Sind die Farbdomänen alle endlich, so folgt aus der Beschränktheit von $\pi(D)$ auch die von D.*

Beweis: Aus der Beschränktheit von $\pi(D)$ folgt die Endlichkeit von $RS(\pi(D), \pi(M_0))$. Es gibt also nur endliche viele erreichbare Markierungen m. Nach Theorem 3 gilt $\pi(RS(D, M_0)) \subseteq RS(\pi(D), \pi(M_0))$ und $\pi(RS(D, M_0))$ ist eine endliche Menge. Da alle $[\![d(p)]\!]_A$ endlich sind, gibt es zu jedem m nur endlich viele Verteilungen der $M(p)(c)$, so dass $\pi(M) = \sum_{c \in [\![d(p)]\!]_A} M(p)(c) = m$ ist. Also ist für alle m die Menge $\{M \mid \pi(M) = m\}$ endlich. Also gibt es zu jedem $m \in \pi(RS(D, M_0))$ nur endliche viele Elemente in $RS(D, M_0)$, so auch $RS(D, M_0)$ endlich ist. Also ist D beschränkt. q.e.d.

2.1.7 Prozesse von Dienstnetzen

Prozesse von algebraischen Netzen – und damit auch von Dienstnetzen – ergeben sich analog zu Prozessen von P/T-Netzen. Ein Prozess ist ein Kausalnetz $K = (B, E, \lessdot)$ zusammen mit dem mit einem Abbildungspaar $\phi = (\phi^P : B \to P, \phi^T : E \to T)$, das jeder Bedingung von K eine Stelle des Netzes N zuweist und jedem Ereignis eine Transition. Daneben wird noch mit $\phi^A : B \to A$ jeder Bedingung b ein Wert der Algebra zugewiesen und durch $\phi^\alpha : E \to (X \to A)$ jedem Ereignis seine Bindung.

Jeder Bedingungsmenge $Q \subseteq B$ wird durch ϕ^A und ϕ^P in natürlicher Weise eine eine Multimenge $\phi^A(Q) : P \to MSA$ zugeordnet, die jeder Stelle eine Multimenge von Farben zuweist. Sie ist für alle $p \in P$ und $c \in A$ definiert durch:

$$\phi^A(Q)(p)(c) := |\{b \in Q \mid \phi^B(b) = p \wedge \phi^A(b) = c\}| \tag{2.7}$$

Damit ein Ereignis $e \in E$ das Schalten der Transition $t = \phi^T(e)$ wiederspiegelt, muss dessen Guard $G(t)$ in der Bindung $\alpha = \phi^\alpha(e)$ gültig sein, und die Multimengen $\phi^A(e^\bullet)(p)$ und $\phi^A({}^\bullet e)(p)$ müssen für alle Stellen p den Kantengewichtungen $\alpha(W(p,t))$ bzw. $\phi^\alpha(e)(W(t,p))$ entsprechen.

Definition 12 *Ein Prozess des Dienstnetzes $D = (N, A, d, r, W, G, M_0)$ ist das Tupel (K, ϕ) mit $K = (B, E, \prec)$ und $\phi = (\phi^P, \phi^T, \phi^A, \phi^\alpha)$, wobei gilt:*

1. *(K, ϕ^P, ϕ^T) ist ein Prozess von N.*
2. *Der Prozess beschreibt den Anfangszustand:*
   $M_0(p)(c) = |\{b \in {}^\circ K : \phi^P(b) = p \phi^A(b) = c\}|$ für alle $p \in P$ und $c \in A$.
3. *$\phi^A : B \to A$ weist jeder Bedingung einen Wert der Algebra zu.*
4. *$\phi^\alpha : E \to (X \to A)$ weist jedem Ereignis seine Bindung zu.*
5. *Die Netzstruktur beschreibt das Schalten von $(\phi^T(e), \phi^\alpha)$:*

$$\begin{aligned} \forall e \in E: \quad & A, \phi^\alpha(e) \models G(\phi^T(e)) \wedge \\ & \forall p \in {}^\bullet\phi^T(e) : \phi^A({}^\bullet e)(p) = \phi^\alpha(e)(W(p, \phi^T(e))) \wedge \\ & \forall p \in \phi^T(e)^\bullet : \phi^A(e^\bullet)(p) = \phi^\alpha(e)(W(\phi^T(e), p)) \end{aligned}$$

Die Menge aller Prozesse von D wird mit $Proc(D)$ bezeichnet.

2.2 Korrektheit von Dienstnetzen

Wir definieren nun die Korrektheit eines Dienstnetzes. Intuitiv ist ein Dienst korrekt, wenn er erstens bei Termination keine unerledigten Aufgaben mehr vorweist, zweitens stets die Möglichkeit hat, einen begonnenen Ablauf zu einem terminierenden fortzusetzen und drittens keine nutzlosen Tätigkeiten spzifiziert.

Definition 13 *Ein Dienstnetz D heißt* korrekt*, wenn gilt:*

1. *Termination: Wird die finale Markierung übertroffen, so wird sie exakt erreicht.*
   $$\forall M \in RS(D, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f$$
2. *Fortsetzbarkeit: Von jeder aus der initialen Markierung erreichbaren Markierung ist die finale Markierung erreichbar.*
   $$\forall M \in RS(D, M_0) : \exists M' \in RS(D, M) : \pi(M') = m_f$$
3. *Aktivierbarkeit: Jede beliebige Transition ist für eine Anfangsmarkierung aktivierbar.*
   $$\forall t \in T : \exists M_0 : \pi(M_0) = m_0 \wedge \exists M \in RS(D, M_0) : M \xrightarrow{t}$$

Ein Dienstnetz D heißt strukturell korrekt*, wenn es für alle Anfangsmarkierungen M mit $\pi(M) = m_0$ korrekt ist.*

Wir betrachten nun die Eigenschaften von Dienstnetzen. Uns interessieren Antworten auf die folgenden Fragen: Können alle möglichen Ausführungen terminieren? In welchen Anfangsmarkierungen kann das Netz terminieren? Falls die Termination nicht garantiert ist, existiert dann eine Kontrollstrategie, so dass das so gesteuerte Netz stets terminiert? Diese Fragen wollen wir beantworten, indem wir die Verwandschaft der Dienstnetze zu den Worklflownetzen ausnutzen.

2.2.1 Relation von Dienst- zu Workflownetzen

Petrinetze sind ein etablierter Formalismus, um Abläufe zu beschreiben. Ein intensiv erforschter Anwendungsbereich stellen die Workflownetze nach (Aalst, 1997, 1998) dar. Workflownetze beschreiben die Abbarbeitung eines Auftrages, wobei der Beginn durch eine Marke auf der Initialstelle i angezeigt wird und die Bearbeitung durch die Markierung, bei der die Finalstelle f eine Marke enthält. Wir betrachten im folgenden die Verwandschaft der Dienstnetze zu Workflownetzen.

Definition 14 *Ein Workflow-Netz ist ein Petrinetz $N = (P, T, F)$, das genau eine Stelle $i \in P$ mit ${}^\bullet i = \emptyset$ und genau eine Stelle $f \in P$ mit $f^\bullet = \emptyset$ besitzt und für das jeder Knoten $n \in (P \cup T)$ auf einem Pfad zwischen i und f liegt.*

Von besonderem Interesse sind solche Netze, bei denen stets gewährleistet ist, dass der Geschätsprozess, egal, was bislang passiert ist, stets noch terminieren kann (angezeigt durch eine Marke auf f) und im Falle der Termination keine überschüssigen Marken mehr im Netz verbleiben. Weiterhin kann man noch fordern, dass jede Transition von der kanonischen Initialmarkeirung $m_0 = i$ aus potentiell aktivierbar ist, denn wäre sie das nicht, dann wäre sie überflüssig, was meist einen Modellierungsfehler darstellt.

Definition 15 *Ein Workflow-Netz $N = (P, T, F)$ heißt* korrekt*, wenn gilt:*

1. *Termination: Wird die finale Markierung übertroffen, so wird sie exakt erreicht.*
$$\forall m \in RS(N, m_0) : m_f \leq m \Longrightarrow m_f = m$$
2. *Fortsetzbarkeit: Von jeder aus der initialen Markierung erreichbaren Markierung ist die finale Markierung erreichbar.*
$$\forall m \in RS(N, m_0) : m_f \in RS(N, m)$$
3. *Aktivierbarkeit: Jede beliebige Transition ist aktivierbar.*
$$\forall t \in T : \exists m \in RS(N, m_0) : m \xrightarrow{t}$$

Korrektheit eines Workflows erfordert, dass von jeder erreichbaren Markierung exakt in m_f terminiert werden kann. Somit ist m_f die einzige Verklemmung. Für potentiell terminierende Workflows ist die Korrektheitbedingung dahingehend abgeschwächt, dass die finale Markierung von der initalen aus erreichbar ist, nicht jedoch unbedingt von jeder von der initalen aus erreichbaren Markierung. Es ist dann im allgemeinen notwendig, das Netz zu steuern (s.u.), um Verklemmungen zu vermeiden.

Die beiden Bedingungen der Termination und der Fortsetzbarkeit (auch als *schwache Korrektheit* bekannt) lassen sich zu einer Bedingungen an die erreichbaren Markierungen zusammenfassen:

$$\forall m \in RS(N, m_0) : (m_f \in RS(N, m)) \wedge (m_f \leq m \Longrightarrow m_f = m)$$

Tabelle 2.1 zeigt einen strukturierte Vorgehensweise zur Erzeugung korrekter Workflows auf. Hierzu wird der Workflow als iterierte Komposition generiert, die entweder sequentiell ($\alpha \cdot \beta$), alternativ ($\alpha + \beta$) oder parallel ($\alpha || \beta$) aus korrekten

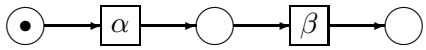
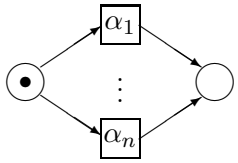
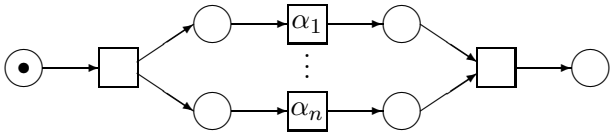
Operator	Petrinetz
Sequenz $\alpha \cdot \beta$	
Alternative (XOR) $\sum_{i=1}^{n} \alpha_i$	
Parallelität (AND) $\Big\|_{i=1}^{n} \alpha_i$	

Tabelle 2.1: Komposition von Workflows

Workflows komponiert werden können. Dabei wird davon ausgegangen, dass die Argumente bereits korrekte Workflows darstellen.

Korrektheit ist eine dynamische Eigenschaft, die nicht monoton ist (van Hee u. a., 2003). Ein Netz, das korrekt für die kanonische Initialmarkierung $m_0 = i$ ist, muss dies nicht für eine andere Markierung sein. Insbesondere kann bereits die Wahl $m = 2i$ die Korrektheit zerstören. So kann das Netz in Abbildung 2.4 von $m = 2 \cdot i$ mit der Schaltfolge ac die Markierung $m' = p_1 + p_2 + p_3 + p_4$ erreichen und dann mit der Transition e die Markierung $m'' = p_1 + p_4 + f$. Das Netz kann also die finale Stelle markieren, ohne dass das Netz geleert hinterlassen würde. Dies Beispiel zeigt auch an, dass man im allgemeinen keinen korrekten Workflow erhält, wenn man eine Transition durch einen korrekten Workflow verfeinert. Das Fehlen der Monotonie motiviert die Verschärfung der Korrektheitsdefinition für eine Vergößerung der Initialmarkierung zu $m = k \cdot i$: Ein Workflow-Netz N heißt *k-korrekt*, wenn von jeder aus der initialen Markierung $m_i = k \cdot i$ erreichbaren Markierung m die finale Markierung $m_f = k \cdot f$ erreichbar ist (Weitere Details finden sich in van Hee u. a., 2003).

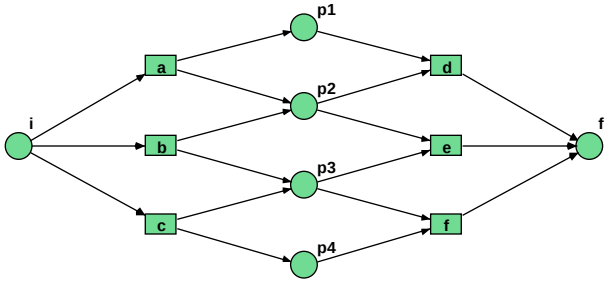


Abbildung 2.4: Ein Workflownetz

Ein Workflow-Netz N hat in der finalen Markierung einen Deadlock. Wir können das Netz jedoch neu starten, indem wir durch eine neu hinzugefügte Transition t_N die Marke in f wieder auf i zurücklegen. Diese Konstruktion wird als Abschluß N^* von N bezeichnet.

Definition 16 *Der Abschluß eines Workflow-Netz $N = (P, T, F)$ ist das Petrinetz $N^* = (P, T \cup \{t_N\}, F \cup \{(o, t_N), (t_N, i)\})$, wobei $t_N \notin (P \cup T)$ gilt.*

Das folgende Theorem zeigt, dass der Korrektheitsbegriff der Workflows eng mit klassischen Netzeigenschaften verbunden ist (vgl. Theorem 11 in Aalst, 1997).

Theorem 5 *Ein Workflow-Netz N ist genau dann korrekt, wenn sein Abschluß N^* in der Initialmarkierung m_0 beschränkt und lebendig ist.*

2.2.2 Reduktion von Ablaufnetzen auf Workflows

Aus der Definition folgt, dass Workflow-Netze spezielle Ablaufnetze sind, bei denen $|{}^\circ N| = |N^\circ| = 1$ gilt. Umgekehrt kann jedes Ablaufnetz durch das Hinzufügen einer intialen und einer finalen Transition in ein Workflow-Netz überführt werden. In ähnlicher Weise behandeln wir Dienstnetze, indem wir alle am Anfang alle Markierungen der minimalen Randstellen zu einem Tupel kombinieren, das die neue Initialmarkierung auf der Stelle i darstellt. Diese Marke wird durch t_i auf die minimalen Randstellen verteilt. Umgekehrt kombiniert t_f alle Werte auf den maximalen Randstellen zu einem Tupel. Die Rolle der neuen Transitionen ist beliebig und wird auf die Spezialrolle r_* festgelegt.

Definition 17 *Sei $N = (P, T, F)$ ein Ablaufnetz und sei $\{i, f\}$, $\{t_i, t_f\}$, P und T paarweise disjunkt, dann ist $\widetilde{N}$ das Netz:*

$$\widetilde{N} := (P \uplus \{i, f\}, T \uplus \{t_i, t_f\}, \widetilde{F})$$

mit $\widetilde{F} = F \cup \{(i, t_i), (t_f, f)\} \cup (\{t_i\} \times {}^\circ N) \cup (N^\circ \times \{t_f\}))$.

Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz und sei ${}^\circ N = \{p_1, \ldots, p_n\}$ und $N^\circ = \{p'_1, \ldots, p'_{n'}\}$, dann ist der Abschluß von D definiert als der Dienst:

$$\widetilde{D} = (\widetilde{N}, A, \tilde{d}, \tilde{r}, \tilde{W}, \tilde{G}, \tilde{M_0})$$

Dabei erweitern wir die Abbildungen folgendermaßen:

$$\begin{aligned}
\tilde{d}(i) &= d(p_1) \times \cdots \times d(p_n) \\
\tilde{d}(f) &= d(p'_1) \times \cdots \times d(p'_{n'}) \\
\tilde{r}(t_i) &= \tilde{r}(t_f) = r_* \\
\tilde{W}(t_i, p) &= x_p = \tilde{W}(p', t_f) \\
\tilde{W}(i, t_i) &= (x_{p_1}, \ldots, x_{p_n}) \\
\tilde{W}(t_f, f) &= (x_{p'_1}, \ldots, x_{p'_{n'}}) \\
\tilde{G}(t_i) &= \tilde{G}(t_f) = \textsc{True} \\
\tilde{M_0}(i) &= (M_0(p_1), \ldots, M_0(p_n))
\end{aligned}$$

Mit Hilfe dieser Konstruktion kann jedes Dienstnetz auf ein Workflownetz reduziert werden.

Theorem 6 *Ist N ein Ablaufnetz, dann ist $\widetilde{N}$ ein Workflownetz.*
Ist D ein Dienstnetz, dann ist $\pi(\widetilde{D})$ ein Workflownetz.

Beweis: Folgt direkt aus den Definitionen 1, 10 und 14. q.e.d.

Im Vergleich mit D hat $\widetilde{D}$ nur zwei Gruppen zusätzlicher erreichbare Markierungen, nämlich die Initialmarkierung M_0 und die neuen Finalmarkierungen M_f, die $\pi(M_f) = m_f$ erfüllen. Analog hat $\widetilde{N}$ als korrekte Workflow im Vergeleich mit $N = \pi(D)$ nur m_i und m_f als zusätzliche Markierungen. Es ist daher nicht verwunderlich, dass beide bezüglich der Korrektheit äquivalent sind:

Theorem 7 *Ein Dienstnetz D ist genau dann korrekt, wenn das Workflownetz $\widetilde{D}$ dies ist.*

Beweis: Zunächst einmal stellen wir fest, dass alle Schaltfolgen in $\widetilde{D}$ die folgende Form haben:

$$\widetilde{M_0} \xrightarrow[\widetilde{D}]{t_i} M_0 \xrightarrow[D]{w} M \quad \text{bzw.} \quad \widetilde{M_0} \xrightarrow[\widetilde{D}]{t_i} M_0 \xrightarrow[D]{w} M \xrightarrow[\widetilde{D}]{t_f} M', \text{ falls } \pi(M) = m_f$$

Also gilt: $M \in RS(D, M_0)$ impliziert $M \in RS(\widetilde{D}, \tilde{M}_0)$.

Für Korrektheit ist die Äquivalenz der Termination und Fortsetzbarkeit in D und $\widetilde{D}$ zu zeigen.

$\Longrightarrow$ Es gelte in D Termination und Fortsetzbarkeit:

$$\forall M \in RS(D, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f$$
$$\forall M \in RS(D, M_0) : \exists M' \in RS(D, M) : \pi(M') = m_f$$

Dann ist für $\widetilde{D}$ analog zu zeigen:

$$\forall M' \in RS(\widetilde{D}, \widetilde{M_0}) : \pi(M') \geq \tilde{m}_f \Longrightarrow \pi(M') = \tilde{m}_f$$
$$\forall M'' \in RS(\widetilde{D}, \widetilde{M_0}) : \exists M''' \in RS(\widetilde{D}, M'') : \pi(M''') = \tilde{m}_f$$

a) Termination: Sei $M' \in RS(\widetilde{D}, \widetilde{M_0})$ eine beliebige Markierung, die $\pi(M') \geq \tilde{m}_f$ erfüllt. Jetzt impliziert $\pi(M') \geq \tilde{m}_f$, dass t_f geschaltet hat. Die Vorgängermarkierung M'' von M' ist aber in D final, erfüllt also $\pi(M'') \geq m_f$, woraus $\pi(M'') = m_f$ folgt, was wiederum $\pi(M') = \tilde{m}_f$ impliziert. Termination gilt also.

b) Fortsetzbarkeit: Ist M in D erreichbar, so auch in $\widetilde{D}$. Zu M existiert $M' \in RS(D, M)$ mit $\pi(M') = m_f$, das auch in $\widetilde{D}$ erreichbar ist. Schalten wir dann t_f in $\widetilde{D}$, so haben wir $\tilde{m}_f$ erreicht.

$\Longleftarrow$ Sei nun umgekehrt in $\widetilde{D}$ Termination und Fortsetzbarkeit gültig, dann ist dies auch für D zu zeigen:

a) Termination: Gilt $\pi(M') \geq \tilde{m}_f \Longrightarrow \pi(M') = \tilde{m}_f$, dann hat t_f geschaltet und die Vorgängermarkierung wurde exakt erreicht.

b) Fortsetzbarkeit: Ist M'' in $\widetilde{D}$ erreichbar, so markiert entweder M'' nur Stellen von D oder $M'' = \widetilde{M_0}$ oder M'' markiert auch f. Im ersten Fall kann stets ein (M''') mit $\pi(M''') = \tilde{m}_f$ erreicht werden, was impliziert, dass zuvor ein (M') mit $\pi(M') = m_f$ erreicht wurde. Im zweiten Fall kann M_0 erreicht werden und davon ein (M') mit $\pi(M') = m_f$. Im dritten Fall darf die Markierung M''' wegen der Gültigkeit der Terminationsbedingung sogar nur f markieren, so dass $\pi(M'') = \tilde{m}_f$ gilt, was impliziert, dass für die Vorgängermarkierung M'''' von M'' auch $\pi(M'''') = m_f$ gilt.

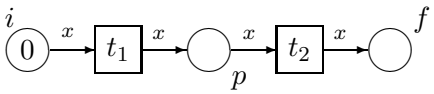


Abbildung 2.5: Ein beschränktes algebraisches Netz, dessen Abschluss nicht beschränkt ist

q.e.d.

Analog zu den Workflownetzen besteht eine enge Verbindung zwischen der Korrektheit des Dienstnetzes D und der Beschränktheit und Lebendigkeit des Abschlusses $\widetilde{D}^*$.

Theorem 8 *Ist $\widetilde{D}^*$ beschränkt und lebendig, dann ist das Dienstnetz D korrekt.*

Beweis: Ist $\widetilde{D}^*$ in M_0 beschränkt und lebendig, dann ist insbesondere t_N lebendig, und wir erreichen von jeder Markierung aus stets eine Markierung M, die t_N aktiviert, also: $\pi(M) \geq m_f$. Dann muss aber auch $\pi(M) = m_f$ gelten, denn ansonsten könnte das Netz nicht beschränkt sein. Also gilt Fortsetzbarkeit und Termination. Beachten wir, dass t_N jede beliebige Anfangsmarkierung M_0' mit $\pi(M_0) = m_0$ generiert werden kann, so können wir aus der Lebendigkeit folgern, dass es für jede Transition t eine Anfangsmarkierung gibt, so dass t aktiviert werden kann. Also gilt die Aktivierungsbedingung. q.e.d.

Man beachte, dass die Beschränktheit für $\widetilde{D}^*$ gefordert wird und nicht für $\widetilde{D}$, denn es ist selbst für korrekte Netze nicht möglich, von der Beschränktheit von D auf die von D^* zu schließen. Dies gilt sogar in struktureller Formulierung: Sei $\widetilde{D}$ ein korrektes Dienstnetz, dass für alle M_0 mit $\pi(M_0) = m_0$ beschränkt ist, dann gilt im allgemeinen nicht auch noch, dass auch $\widetilde{D}^*$ beschränkt ist. Abbildung 2.5 zeigt ein korrektes, für alle M_0 beschränktes Dienstnetz, für das $\widetilde{D}^*$ nicht beschränkt ist, da die Stelle p jeden beliebigen Wert annehmen kann, da die Transition t_N eine beliebige Zahl als Anfangsmarkierung generieren kann.

Von der Korrektheit können wir auf die Lebendigkeit des Dienstnetzes schließen, nicht aber auf die Beschränktheit, denn das Dienstnetz aus Abbildung 2.2 ist korrekt und unbeschränkt. Die Umkehrung gilt daher nur in folgender eingeschränkter Form.

Theorem 9 *Wenn das Dienstnetz D strukturell korrekt ist, dann ist $\widetilde{D}^*$ für alle Markierungen M_0 mit $\pi(M_0) = m_0$ lebendig.*

Beweis: Betrachten wir ein strukturell korrektes Dienstnetz in einer Markierungen M_0 mit $\pi(M_0) = m_0$. Ist das Netz strukturell korrekt, so ist mit der Fortsetzbarkeitseigenschaft von jeder erreichbaren Markierung $M \in RS(D, M_0)$ die finale Markierung M_f mit $\pi(M_f) = m_f$ erreichbar – unabhängig von der Wahl für M_0. Also ist t_N aktivierbar. Um zu zeigen, dass jede Transition t aktiviert werden kann, nutzen wir aus, dass t_N jede beliebige Anfangsmarkierung M_0' mit $\pi(M_0') = m_0$ generieren kann, und da D strukturell korrekt ist, gilt die Aktivierungsbedingung auch für M_0'. Die Aktivierungsbedingung garantiert, dass jede Transition t für mindestens eine Anfangsmarkierung $M_{0,t}$ aktiviert werden kann. Da diese von t_N generiert werden kann, ist $\widetilde{D}^*$ lebendig. q.e.d.

Wenn das Dienstnetz D korrekt oder strukturell korrekt ist, dann muss $\widetilde{D}^*$ jedoch nicht beschränkt sein, wie das unbeschränkte, aber strukturell korrekte Netz in Abbildung 2.2 zeigt.

Betrachten wir nur beschränkte Dienstnetze, so können wir Theorem 8 und 9 folgendermaßen zusammenfassen:

Theorem 10 *Sei $\widetilde{D}$ ein Dienstnetz, das für alle M_0 mit $\pi(M_0) = m_0$ beschränkt ist, dann gilt: Das Dienstnetz D ist genau dann strukturell korrekt, wenn $\widetilde{D}^*$ für alle Markierungen M_0 mit $\pi(M_0) = m_0$ lebendig ist.*

Beweis: Das Dienstnetz $\widetilde{D}$ ist genau dann strukturell korrekt, wenn D dies ist. Sei $\widetilde{D}$ für jedes M_0 mit $\pi(M_0) = m_0$ beschränkt.

Wenn das Dienstnetz $\widetilde{D}$ strukturell korrekt ist, dann ist es für alle M_0 mit $\pi(M_0) = m_0$ korrekt und damit ist $\widetilde{D}^*$ nach Theorem 9 auch für alle M_0 lebendig.

Für jedes beliebige M_0 mit $\pi(M_0) = m_0$ folgt aus der Lebendigkeit (zusammen mit der gegebenen Beschränktheit) in M_0 aus Theorem 8 die Korrektheit. Gilt Lebendigkeit für alle diese M_0, so folgt die Korrektheit für alle M_0, also die strukturelle Korrektheit. q.e.d.

2.3 Rollenkomponenten von Dienstnetzen

Die Definition von Workflownetzen hat also mit den Stellen i und f einen globalen Start- bzw. Endpunkt der Bearbeitung. Workflownetze modellieren somit eine zentralisierte Sichtweise auf den Geschätsprozess. Ist dagegen (wie in Abbildung 2.6) eine verteilte Bearbeitung durch verschiedene interagierende Akteure zu modellieren, so ist es notwendig, das System auf mehrere Anfangs- bzw. Endstellen zu verallgemeinern. Dies vereinfacht auch die Definition der Komposition von Abläufen.[3]

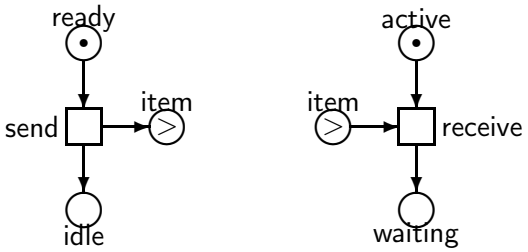


Abbildung 2.6: Producer/Consumer-Interaktion

Für zwei einfache Netze $N_i = (P_i, T_i, F_i), i = 1, 2$ ist die Komposition als komponentenweise Vereinigung zu definieren:

$$(N_1 \| N_2) = (P_1 \cup P_2, T_1 \cup T_2, F_1 \cup F_2)$$

[3]Komposition von Modulen zu größeren Einheiten wird an vielen Stellen bei der Softwareentwicklung genutzt. Insbesondere ein „bottom-up“ Vorgehen, das durch die Objektorientierung propagiert wird, besitzt als zentrales Element die Komposition von einfachen zu komplexen Objekten. Die Interaktion von vielen, vorgefertigten Objekten ist unter dem Begriff der Komponenten-Architekturen bekannt geworden (Griffel, 1998).

Eine Komposition bezüglich gemeinsamer Stellen beschreibt eine asynchrone Kopplung der Systeme. Stellenfusion modelliert den gemeinsamen Zugriff zweier Komponenten auf ein geteiltes Objekt, z.B. auf einen Nachrichtenpuffer oder eine gemeinsam benutzte Ressource. Wenn die Transitionsmengen disjunkt sind, schränkt das Modell keine Fortsetzbarkeitseigenschaften ein, so dass die Kopplung als *schwache Komposition* bezeichnet wird. Für ein schwachgekoppeltes System übertragen sich alle Fortsetzbarkeitseigenschaften der Einzelkomponenten auf das Gesamtsystem. Die Eigenschaft der Fortsetzbarkeit ist gleichbedeutend mit der Forderung, dass das Transitionssystem der Komposition $N_1 \| N_2$ in der Einschränkung auf die Aktionen von N_1 das Transitionssystem von N_1 enthält. Eine Komposition fügt demnach nur Übergänge zwischen Zuständen hinzu. Es ist intuitiv klar, dass sich alle Fortsetzbarkeitseigenschaften der Einzelkomponenten auf ein schwachgekoppeltes System übertragen.

Neben der Verschmelzung von Stellen, welche die einfachste Modellierung einer asynchronen Kopplung darstellt, existiert das – im folgenden nicht verwendete – Konzept der Transitionsverschmelzung. Eine Transitionsverschmelzung beschreibt eine synchrone Aktion. Transitionverschmelzung koppelt zwei Systeme eng aneinander, der Raum der erreichbaren Zustände wird sowohl eingeschränkt als auch erweitert. Er wird eingeschränkt, indem Zustandsübergänge der Komponente N_1 direkt an einen Übergang von N_2 gebunden werden können. Ist dieser nicht möglich, so wird der Übergang auch in N_1 nicht mehr möglich sein. Er wird erweitert, indem N_2 durch Hinzufügen von Transitionen den Erreichbarkeitsgraph von N_1 erweitern kann.

Ebenso können wir Komposition $(D_1 \| D_2)$ zweier Dienstnetze $D_i = (N_i, A, d_i, r_i, W_i, G_i, M_i), i = 1, 2$ über der gleichen Algebra definieren. Dabei ist für die Komponierbarkeit zu fordern, dass die Anschriften auf den gemeinsamen Elementen übereinstimmen:

$$
\begin{aligned}
&d_1(p) = d_2(p) \text{ und } M_1(p) = M_2(p) \text{ für alle } p \in P_1 \cap P_2 \\
&r_1(t) = r_2(t) \text{ und } G_1(t) = G_2(t) \text{ für alle } t \in T_1 \cap T_2 \\
&W_1(f) = W_2(f) \text{ für alle } f \in F_1 \cap F_2
\end{aligned}
$$

Unter dieser Vorraussetzung ergibt sich die Komposition wie folgt:

$$(D_1 \| D_2) = (N_1 \| N_2, A, d_1 \cup d_2, r_1 \cup r_2, W_1 \cup W_2, G_1 \cup G_2, M_1 + M_2)$$

2.3.1 Netzkomponenten

In der folgenden Definition orientieren wir uns an der Komponentendefinition in (Kindler, 1997), bei denen Schnittstellen nur einseitig genutzt werden. Sie sind also entweder nur zum Lesen (Input) oder nur zum Schreiben (Output). Wir lassen im Vergleich zur Orginaldefinition hier die Spezifikation der Transitionen mit Fortschrittseigenschaft weg, da sie für uns nicht wichtig sind.

Definition 18 *Eine* Netzkomponente $\Gamma = (N, m_0, I, O)$ *besteht aus einem folgenden Komponenten:*

1. $N = (P, T, F)$ *ist ein* T*-schlichtes Petrinetz mit Markierung* $m_0 : P \to \mathbb{N}$.
2. $I, O \subseteq P$ *sind Mengen an Input- bzw. Output-Schnittstellen mit* ${}^\bullet T \cap O = T^\bullet \cap I = \emptyset$ *und* $m_0(p) = 0$ *für alle* $p \in I \cup O$.

Eine Komponente heißt geschlossen, *wenn* $I \cup O = \emptyset$ *gilt.*

Zwei Komponenten Γ_1 *und* Γ_2 *sind komponierbar, wenn* $(T_1 \cap T_2) = \emptyset$ *und* $(P_1 \cap P_2) \subseteq ((I_1 \cap O_2) \cup (I_2 \cap O_1))$ *gilt.*

Die Komposition zweier komponierbarer Komponenten ist:

$$\Gamma_1 \parallel \Gamma_2 := (N_1 \cup N_2, m_1 + m_2, I, O)$$

mit $I = ((I_1 \setminus O_2) \cup (I_2 \setminus O_1))$ *und* $O = ((O_1 \setminus I_1) \cup (O_2 \setminus I_1))$.

Die Komposition zweier Komponenten ist wiederum eine. Offensichtlich ist die Komposition assoziativ und kommutativ, da die Mengenvereinigung dies ist.

Damit Prozesse definierbar sind, fordert Kindler, dass N auch T*-schlicht* ist, d.h. Transitionen sind nicht isoliert: $\forall t \in T : {}^\bullet t \neq \emptyset \wedge t^\bullet \neq \emptyset$. Prozesse einer Komponenten werden analog zu Prozessen von P/T-Netzen definiert. Die Definition unterscheidet jedoch zwischen den Ereignissen der Komponenten selbst und solchen, die die Umgebung erzeugt. In der folgenden Definition werden nur die Bedingungen abgebildet, während die Ereignisse implizit bleiben. Dies hat den Vorteil, dass externe Aktionen, deren Name unbekannt ist, nicht benannt werden müssen. Es wird davon ausgegangen, dass eine globale Obermenge $\mathcal{P}$ der Stellen existiert, damit die Komponenten eingebettet werden können.

Definition 19 *Sei* $\Gamma = (N, m_0, I, O)$ *eine Komponente mit* $N = (P, T, F)$. *Sei* $K = (B, E, \lessdot)$ *ein vorgängerendliches Kausalnetz und* $\phi : B \to \mathcal{P}$ *eine Abbildung, dann ist* (K, ϕ) *ein* Prozess *der Komponenten* Γ*, falls die folgenden Eigenschaften gelten:*

1. *Die minimalen Stellen* ${}^\circ K$ *beschreiben die Anfangsmarkierung:* $\phi({}^\circ K)|_P = m_0$.
2. *Für jedes Ereignis* $e \in E$ *gilt eine der beiden Bedingungen:*
   a) *Interne Aktion: Es existiert eine Transition* $t \in T$*, so dass* $\phi({}^\bullet e) = {}^\bullet t$ *und* $\phi(e^\bullet) = t^\bullet$.
   b) *Externe Aktion: Für alle Stellen* $x \in \mathcal{P}$ *gilt* $\phi({}^\bullet e)(x) \leq 1$ *und* $\phi(e^\bullet)(x) \leq 1$. *Aus* $\phi({}^\bullet e)(x) = 1$ *folgt* $x \in O \cup (\mathcal{P} \setminus P)$ *und* $\phi(e^\bullet)(x) = 1$ *impliziert* $x \in I \cup (\mathcal{P} \setminus P)$.

Die Menge aller Prozesse von Γ *wird mit* $Proc(\Gamma)$ *bezeichnet.*

Man beachtet, dass nach Bedingung (1) die minimalen Stellen ${}^\circ R$ auch andere Stellen beschreiben können, d.h. eine Bedingung $b \in {}^\circ R$ kann Stellen außerhalb der Komponenten beschreiben: $\phi(b) \in \mathcal{P} \setminus P$.

Mit dieser Definition der Komponentensemantik entspricht das Verhalten der Komposition $\Gamma_1 \parallel \Gamma_2$ dem Durchschnitt der Einzelverhalten (vgl. Kindler, 1997, Proposition 10).

Theorem 11 *Seien* Γ_1, Γ_2 *zwei komponierbare Komponenten, dann gilt:*

$$Proc(\Gamma_1 \parallel \Gamma_2) = Proc(\Gamma_1) \cap Proc(\Gamma_2)$$

Dieses Theorem charakterisiert die Semantik der Komponenten als kompositional, d.h. die Semantik der Komposition ergibt sich direkt als der Durchschnitt der Semantiken der Komponenten.

2.3.2 Rollenkomponenten von Diensten

Sei $N = (P, T, F)$ das dem Dienstnetz zugrundeliegende Netz und sei $R \subseteq R(N)$ eine Teilmenge der im Netz vorkommenden Rollen. Dann können wir das Netz auf den Teil einschränken, der alle R zugeordneten Transitionen und deren umliegende Stellen enthält. Das resultierende Netz bezeichnen wir als R-Komponente. Rollenkomponenten stellen den Dienst dar, der von der Rolle R erbracht wird. Sie stellen implizit eine Spezifikation der Rechte und der Pflichten, die mit dieser Rolle verbunden sind, dar.

Der Vorteil von Rollenkomponenten von Diensten gegenüber beliebigen Komponenten ist, dass wir in voraus wissen, welche Komponenten auftreten können.

Definition 20 *Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienst mit $N = (P, T, F)$ und sei $R \subseteq R(D)$.*

- *Die* R-Komponente *von N ist das durch $T_R := r^{-1}(R)$ und $P_R := ({}^\bullet T_R \cup T_R{}^\bullet)$ induzierte Teilnetz von N, das mit $N[R]$ bezeichnet wird.*
- *Jedes Teilnetz $N' = (P', T', F')$ von N induziert das Dienstnetz:*

$$D|_{N'} = (N', A, d|_{P'}, r|_{T'}, W|_{F'}, G|_{T'}, M_0|_{P'}) \tag{2.8}$$

- *Ist $N' = N[R]$ speziell eine Rollenkomponente, dann ist $D[R] := D|_{N'}$ die* R-Komponente *von D.*

Sei $\mathcal{E} = \{R_1, \ldots, R_n\}$ eine Mengenpartition der Rollen $R(D)$ eines Dienstnetz D. Sie erzeugt eine Menge an Rollenkomponenten: $D[R_1], \ldots, D[R_n]$. Für den Spezialfall $\mathcal{E} = \{R, \bar{R}\}$ für ein beliebiges $R \subseteq R(N)$ und mit $\bar{R} := R(N) \setminus R$ sprechen wir für die Aufteilung von $N = N[R] \cup N[\bar{R}]$ kurz von der R-Zerlegung von N.

Wir definieren eine Komposition auf Dienstnetzen unter der Voraussetzung, dass die Dienste disjunkte Transitionsmengen haben, als komponentenweise Vereinigung.

Theorem 12 *Sei D ein Dienst und sei $R_1, R_2 \subseteq R(D)$ mit $R_1 \cap R_2 = \emptyset$, dann gilt:*

$$N[R_1 \cup R_2] = N[R_1] \| N[R_2]$$

Beweis: Da R_1 und R_2 disjunkt sind, sind auch $T_{N[R_1]}$ und $T_{N[R_2]}$ disjunkt. Es gilt:

$$\begin{aligned} T_{N[R_1 \cup R_2]} &= r^{-1}(R_1 \cup R_2) \\ &= r^{-1}(R_1) \cup r^{-1}(R_2) \\ &= T_{N[R_1]} \cup T_{N[R_2]} \\ &= T_{N[R_1] \| N[R_2]} \end{aligned}$$

Außerdem ist:

$$\begin{aligned} P_{N[R_1 \cup R_2]} &= {}^\bullet T_{N[R_1 \cup R_2]} \cup T_{N[R_1 \cup R_2]}{}^\bullet \\ &= ({}^\bullet T_{N[R_1]} \cup {}^\bullet T_{N[R_2]}) \cup (T_{N[R_1]}{}^\bullet \cup T_{N[R_2]}{}^\bullet) \\ &= ({}^\bullet T_{N[R_1]} \cup T_{N[R_1]}{}^\bullet) \cup ({}^\bullet T_{N[R_2]} \cup T_{N[R_2]}{}^\bullet) \\ &= P_{N[R_1]} \cup P_{N[R_2]} = P_{N[R_1] \| N[R_2]} \end{aligned}$$

Da die Stellen und Transition übereinstimmen, generieren $N[R_1 \cup R_2]$ und $N[R_1] \| N[R_2]$ die gleichen Teilnetze. q.e.d.

Die R-Komponenten $D[R_i]$ beschreiben den Dienst D vollständig, in dem Sinne, dass wir das Netz aus seinen Komponenten rekonstruieren können: $N = N[R_1] \| \cdots \| N[R_n]$.

Theorem 13 *Sei D ein Dienstnetz und $N = \pi(D)$ ohne isolierte Stellen. Für jede Mengenpartition $\mathcal{E}$ der Rollen $R(D)$ gilt:*

$$N = \Big\|_{R\in\mathcal{E}} N[R] \quad \textit{und} \quad D = \Big\|_{R\in\mathcal{E}} D[R]$$

Beweis: Wir zeigen zunächst $N = \|_{R\in\mathcal{E}} N[R]$. Da $\mathcal{E}$ eine Partition ist, gilt $\bigcup_{R\in\mathcal{E}} R = R(D)$ und damit auch:

$$\bigcup_{R\in\mathcal{E}} T_{N[R]} = \bigcup_{R\in\mathcal{E}} r^{-1}(R) = T$$

Hier gilt sogar, dass alle $T_{N[R]}$ disjunkt sind. Dies ist notwendig, damit die Komposition der $N[R]$ überhaupt definiert ist.

Da N ohne isolierte Stellen ist, gilt ${}^\bullet p \cup p^\bullet \neq \emptyset$, und damit folgt:

$$\bigcup_{R\in\mathcal{E}} P_{N[R]} = \bigcup_{R\in\mathcal{E}} ({}^\bullet T_{N[R]} \cup T_{N[R]}{}^\bullet) = P$$

Die $P_{N[R]}$ sind nicht nowendigerweise disjunkt.

Sei $(p,t) \in F \cap (P \times T)$, dann gibt es genau eine Rolle R, so dass $(p,t) \in F_{N[R]}$, da t in genau einem $N[R]$ vorkommt. Analog für $(t,p) \in F \cap (T \times P)$. Also gilt

$$\bigcup_{R\in\mathcal{E}} F_{N[R]} = F$$

Damit ist $N = \|_{R\in\mathcal{E}} N[R]$ klar, da die Knotenmenge vom N von denen der $N[R]$ überdeckt wird und alle $N[R]$ Teilnetze von N sind.

Da sich $D[R]$ stets als Einschränkung von D ergibt, folgt $D = \|_{R\in\mathcal{E}} D[R]$ aus $N = \|_{R\in\mathcal{E}} N[R]$, denn alle Abbildung auf geteilten Elementen sind konsistent definiert. q.e.d.

Die Teilnetze ergeben somit eine Überdeckung, die aber im allgemeinen nicht disjunkt ist, da die Kommunikationsstellen geteilt werden.

R-Komponenten erweisen sich als Komponenten im Sinne von Definition 18. Dazu definieren wir die Kommunikationskanäle als Schnittstellen. Kommunikationskanäle sind Stellen, die Transitionen unterschiedlicher Rollen verbinden: $R({}^\bullet p) \neq R(p^\bullet)$ gilt.

Die initialen und finalen Stellen eines Dienstes sind leicht zu ermitteln. Stellen sind initial, wenn ${}^\bullet p = \emptyset$ gilt. Sie sind final, wenn $p^\bullet = \emptyset$ gilt. Die Bedingung an die Rollentypisierung

$$\forall p \in P : |R({}^\bullet p)|, |R(p^\bullet)| \leq 1$$

impliziert für Stellen mit ${}^\bullet p, p^\bullet \neq \emptyset$ bereits $\forall t_1, t_2 \in {}^\bullet p : r(t_1) = r(t_2)$ und $\forall t_1, t_2 \in p^\bullet : r(t_1) = r(t_2)$, nicht jedoch $\forall t_1 \in {}^\bullet p : \forall t_2 \in p^\bullet : r(t_1) = r(t_2)$. Gilt die letzte Bedingung, dann verbindet die Stellen zwei Transitionen, die in der gleichen R-Komponenten liegen, die Stelle ist also intern. Gilt die Bedingung nicht, dann liegen die Transitionen in verschiedenen R-Komponenten, die Stelle ist also eine Kommunikationsstelle. Somit können alle Stellen eindeutig als Initial-, Terminal-, Binnen- oder als Kommunikationsstellen identifizert werden.

Definition 21 *Sei D ein Dienst und $N[R] = (P_R, T_R, F_R)$ seine R-Komponente. Definiere $\Gamma_D[R] := (N[R], m_0^R, I^R, O^R)$ mit*

$$\begin{aligned} I^R &= \{p \in P_{KK}(N) \mid p^\bullet \subseteq T_R\} \\ O^R &= \{p \in P_{KK}(N) \mid {}^\bullet p \subseteq T_R\} \\ P_i^R &= \{p \in P \mid R({}^\bullet p) = \emptyset \wedge p^\bullet \subseteq T_R\} \\ P_f^R &= \{p \in P \mid R(p^\bullet) = \emptyset \wedge {}^\bullet p \subseteq T_R\} \\ m_0^R(p) &= 1, \textit{ falls } p \in P_i^R \textit{ und } 0 \textit{ sonst} \end{aligned}$$

Theorem 14 *Wenn D ein Dienst und $N[R]$ eine R-Komponente ist, dann ist $\Gamma_D[R]$ eine Komponente nach Definition 18.*

Beweis: Es ist $I^R \subseteq P_R$ zu zeigen: Es gilt $I^R \subseteq P_R$, denn $p^\bullet \subseteq T_R$ impliziert $p \in ({}^\bullet T_R \cup T_R{}^\bullet) = P_R$. Analog folgt $O^R \subseteq P_R$, $P_i^R \subseteq P_R$ und $P_i^R \subseteq P_R$.

Wir zeigen $T_R{}^\bullet \cap I^R = \emptyset$: Sei $p \in I^R$, dann gilt nach Definition $R({}^\bullet p) \neq R(p^\bullet) \wedge p^\bullet \subseteq T_R$. Da für Kommunikationsstellen stets $|R({}^\bullet p)| = |R(p^\bullet)| = 1$ gilt, ist $R({}^\bullet p) = \{r_1\}$ und $R(p^\bullet) = \{r_2\}$ für $r_1 \neq r_2$. Da $p^\bullet \subseteq T_R$ gilt, muss $R(p^\bullet) \subseteq R$ sein, d.h. $r_2 \in R$ und $r_1 \notin R$. Aus $r_1 \notin R$ folgt dann ${}^\bullet p \cap T_R = \emptyset$. Dies ist gleichbedeutend mit $(\_ F_R p) = \emptyset$. Also gilt in $N[R]$ auch ${}^\bullet I^R = \emptyset$ und dies ist gleichbedeutend mit $T_R{}^\bullet \cap I^R = \emptyset$. Analog zeigt man $O^R \cap {}^\bullet T_R = \emptyset$.

Damit ist $\Gamma_D[R]$ eine Komponente. q.e.d.

2.3.3 Prozesse für Rollenkomponenten

Da für Komponenten eine Prozessdefinition existiert, können wir daher dies auch auf Rollenkomponenten übertragen:

$$Proc(N[R]) := Proc(\Gamma_D[R]) \tag{2.9}$$

Wir erweitern das Konzept der Komponentenprozesse, die keine feste Umgebung annehmen, nun auf die Komponenten $D[R]$ eines Dienstnetzes. Dabei nutzen wir aus, dass Rollenkomponenten eines Dienstnetzes die Komponente $\Gamma_D[R]$ definiert, die eine Komponente nach Definition 18 ist. Es ist daher naheliegend, die Definition der Komponentenprozesse aus Definition 22 zu verallgemeinern. Dies geschieht analog zur Definition der Dienstprozesse in Definition 12. Dazu nehmen an, dass alle globalen Stellen $p \in \mathcal{P} \setminus P$ auch einen Typ $d(p)$ besitzen.

Definition 22 *Sei D ein Dienstnetz und $R \subseteq R(D)$ eine Rolle. Ein* Prozess von $D[R]$ *ist das Tupel (K, ϕ) mit $K = (B, E, \prec)$ und $\phi = (\phi^P, \phi^A)$, wobei gilt:*

1. *Der Prozess beschreibt in der Einschränkung auf $P \subseteq \mathcal{P}$ den Anfangszustand M_0 von $D[R]$:*

   $M_0(p)(c) = |\{b \in {}^\circ D[R] : \phi^P(b) = p\phi^A(b) = c\}|$ *für alle $p \in P$ und $c \in A$.*

2. *$\phi^A : B \to A$ weist jeder Bedingung einen typgerechten Wert der Algebra zu: $\phi^A(p) \in [\![d(p)]\!]$ für alle $p \in \mathcal{P}$.*

3. *Für jedes Ereignis $e \in E$ gilt eine der beiden Bedingungen:*

a) Interne Aktion: Es existiert eine Transition $t \in T$ und eine Bindung α, so dass $\phi^P({}^\bullet e) = {}^\bullet t$ und $\phi^P(e^\bullet) = t^\bullet$ und e beschreibt für eine Belegung α das Schalten von (t, α):

$$A, \alpha \models G(\phi^T(e)) \quad \wedge \quad \forall p \in {}^\bullet t : \phi^A({}^\bullet e)(p) = \alpha(W(p,t)) \quad \wedge \quad \forall p \in t^\bullet : \phi^A(e^\bullet)(p) = \alpha(W(t,p))$$

b) Externe Aktion: Für alle Stellen $x \in \mathcal{P}$ gilt $\phi({}^\bullet e)(x) \leq 1$ und $\phi(e^\bullet)(x) \leq 1$. Aus $\phi({}^\bullet e)(x) = 1$ folgt $x \in O \cup (\mathcal{P} \setminus P)$ und $\phi(e^\bullet)(x) = 1$ impliziert $x \in I \cup (\mathcal{P} \setminus P)$.

Die Menge aller Prozesse von $D[R]$ wird mit $Proc(D[R])$ bezeichnet.

Die Prozessdefinition für die gefärbten Dienstnetze ergibt sich als kanonische Erweiterung:

Theorem 15 *Wenn $(K, (\phi^P, \phi^A))$ ein Prozess von $D[R]$ ist, dann ist (K, ϕ^P) eine Komponentenprozess von $\Gamma_D[R]$.*

Beweis: Direkt aus dem Vergleich von Definition 22 mit Definition 19. q.e.d.

Da die Komposition zweier Rollenkomponenten $D[R_1] \| D[R_2]$ erzwingt, dass die Anschriften konsitent sind, können wir ganz analog zu Theorem 11 folgern, dass such das Verhalten der Komposition als der Durchschnitt der Einzelverhalten ergibt:

Theorem 16 *Seien $D[R_1], D[R_2]$ zwei komponierbare Komponenten mit $R_1 \cap R_2$, dann gilt:*

$$Proc(D[R_1] \parallel D[R_2]) = Proc(D[R_1]) \cap Proc(D[R_2])$$

2.3.4 Verfeinerungsstruktur auf Dienstnetzen

Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz. Wir hatten Stellen in (2.5) als Kommunikationskanäle definiert, wenn sie unterschiedliche Rollen verbinden.

Wenn wir uns fragen, ob sich Dienste ähnlich verhalten, dann bezieht sich die Verhaltensähnlichkeit stets auf die Kommunikationen. Da Nachrichten Stellen, also passive Elemente sind, können wir mit ihnen nicht direkt eine Verhaltensähnlichkeit definieren. Dies lässt sich aber leicht erreichen, wenn wir mit jeder Nachricht ein Ereignis assozieren. Dazu verfeinern wir jede Nachricht p, indem wir sie in zwei Stellen p^{in} und p^{out} aufspalten, die durch die Transition t_p verbunden werden. Die Stellen p^{in} und p^{out} haben den gleichen Typ wie p. An den Kanten (p^{in}, t_p) und (t_p, p^{out}) notieren wir jeweils die gleiche Variable x, so dass jede Marke von p^{in} zu p^{out} geschaltet werden kann. Man erkennt sofort, dass bis auf einfache Umbenennungen sich das Verhalten des Netzes durch diese Transformation nicht ändert, denn jede Markierung M spaltet sich für jeden markierten Kommunikationskanal p in die Markierungen M^{in} und M^{out} auf, die im Erreichbarkeitsgraph nur durch die t_p verbunden sind. Da Kommunikationsstellen nie initial sind, sind sie in allen Anfangsmarkierungen unmarkiert, so dass wir die Markierung von p^{in} und p^{out} die leere Multimenge ist. Die Rolle der neuen Transitionen ist beliebig und wird auf die Spezialrolle r_* festgelegt.

Definition 23 *Sei $D = (N, A, d, r, W, G, M_0)$ ein Dienstnetz. Die* Kommunikationserweiterung von D *ist das Dienstnetz:*

$$\hat{D} = (N', A, d', r', W', G', M_0')$$

Dabei ist $N' = (P', T', F')$ und die Abbildungen d, r, W, G, M_0 modifizieren wir folgendermaßen:

$$\begin{aligned}
P' &= (P \setminus P_{KK}(N)) \cup \{p^{in}, p^{out} \mid p \in P_{KK}(N)\} \\
T' &= T \cup \{t_p \mid p \in P_{KK}(N)\} \\
F' &= F \cup \{(p^{in}, t_p), (t_p, p^{out}) \mid p \in P_{KK}(N)\} \\
d'(p^{in}) &= d'(p^{out}) = d(p) \\
r'(t_p) &= r_* \\
W'(p^{in}, t_p) &= x = W'(t_p, p^{out}) \qquad x \in X_{d(p)} \\
M_0'(p^{in}) &= M_0'(p^{out}) = \mathbf{0}
\end{aligned}$$

Wir wollen nun definieren, wann zwei Dienste bezüglich ihres Kommunikationsverhaltens äquivalent sind. Der hier verwendte Äquivalenzbegriff beruht auf dem Konzept der *Bisimulation* (vgl. Park, 1980; Milner, 1989). Zwei Systemzustände P und Q sind bisimilar (notiert als $P \sim Q$), wenn jeder Zustandsübergang $P \xrightarrow{a} P'$ im anderen System durch eine den Zustandsübergang $Q \xrightarrow{a} Q'$ simuliert werden kann und P' und Q' wiederum bisimilar sind. Bisimulation kann durch die Forderung, dass folgendes Diagramm durch die Existenz einer Konfiguration Q' (bzw. P' im symmetrischen Fall) kommutativ ist, dargestellt werden:

$$\begin{array}{ccc} P & \sim & Q \\ a\downarrow & & \downarrow a \\ P' & \sim & Q' \end{array}$$

Haben wir zwei Dienste D_1 und D_2, dann müssen diese, um ähnlich sein zu können, auf jeden Fall die gleichen Kommunikationskanäle besitzen. Daraus folgt, dass beide Dienste die gleichen hinzugefügten Transitionen t_p besitzen. Auf diesen Aktionen definieren wir dann eine Bisimulation $\approx$ bezüglich der Markierungen der beiden Dienste. Hierbei müssen wir von den internen Aktionen der Dienstnetze abstrahieren, d.h. von allen nicht kommunikativen Transitionen. Diese Form der Bisimilarität heißt *Kommunikationsäquivalenz*. Dazu definieren wir den löschenden Homomorphismus h mit:

$$h(t, \alpha) = \begin{cases} (t, \alpha), & \text{falls } t = t_p, p \in P_{KK}(D) \\ \lambda, & \text{sonst} \end{cases}$$

Wir definieren mit Hilfe des Homomorphismus h eine weitere Übergangsrelation $\stackrel{\widehat{\cdot}}{\longrightarrow}$, die beliebig viele interne Schritte vor und nach einer Kommunikationstransition erlaubt:

$$M_1 \xrightarrow[\widehat{D}]{\widehat{t,\alpha}} M_2 :\Longleftrightarrow \exists w : M_1 \xrightarrow[\widehat{D}]{w} M_2 \wedge h(w) = (t, \alpha)$$

Definition 24 *Seien D_1 und D_2 zwei Dienstnetze mit gleichen Kommunikationskanälen, d.h. $P_{KK}(D_1) = P_{KK}(D_2)$.*

Sei M_1 eine Markierungen von D_1 und M_2 eine Markierungen von D_2. M_1 und M_2 sind kommunikationsbisimilar*, notiert als $M_1 \approx M_2$, falls $(M_1, M_2) \in \mathcal{B}$ für eine Kommunikationsbisimulation $\mathcal{B}$ gilt.*

Eine binäre Relation $\mathcal{B}$ auf heißt Kommunikationsbisimulation*, falls für alle $(M_1, M_2) \in \mathcal{B}$ gilt:*

1. *Wenn $M_1 \xrightarrow[\hat{D}_1]{\widehat{t,\alpha}} M_1'$ gilt, dann existiert ein M_2', so dass $M_2 \xrightarrow[\hat{D}_2]{\widehat{t,\alpha}} M_2'$ und $(M_1', M_2') \in \mathcal{B}$ gilt.*

2. *Wenn $M_2 \xrightarrow[\hat{D}_2]{\widehat{t,\alpha}} M_2'$ gilt, dann existiert ein M_1', so dass $M_1 \xrightarrow[\hat{D}_1]{\widehat{t,\alpha}} M_1'$ und $(M_1', M_2') \in \mathcal{B}$ gilt.*

Man erkennt leicht, dass $\approx$ eine Äquivalenzrelation ist. Man beachte, dass für die Definition der Simulation die erweiterten Netze $\hat{D}_i, i = 1, 2$ und nicht die Dienstnetze D_i selbst verwendet werden.

Mit Hilfe der Kommunikationsbisimulation definieren wir nun die Rollensubstitution:

$$\langle D; \{(R_1, R_1'), \ldots, (R_n, R_n')\}; D' \rangle$$

Sie drückt aus, dass sich dass Verhalten nicht ändert, wenn wir in D eine R_i-Komponente $D[R_i]$ gegen die korrespondierende R_i'-Komponente $D'[R_i']$ von D' substituieren.

Gilt $n = 1$, dann ist $\langle D; \rho; D' \rangle$ von der Form $\langle D; \{(R, R')\}; D' \rangle$. Dies impliziert $R = R(D)$ und $R' = R(D')$.

Definition 25 *Die* Verhaltensäquivalenz $\langle D; \rho; D' \rangle$ *gilt genau dann, wenn folgendes erfüllt ist:*

1. $\mathcal{E} = \{R_1, \ldots, R_n\}$ *ist eine Mengenpartition auf* $R(D)$.

2. $\mathcal{E}' = \{R_1', \ldots, R_n'\}$ *ist eine Mengenpartition auf* $R(D')$.

3. $\rho : \mathcal{E} \to \mathcal{E}'$ *ist eine Bijektion.*

4. *Für jede Auswahl* $A_i, B_i \in \{D[R_i], D'[\rho(R_i)]\}$ *gilt* $(A_1 \| \cdots \| A_n) \approx (B_1 \| \cdots \| B_n)$.

Statt $\langle D; \rho; D' \rangle$ *notieren wir auch explizit:* $\langle D; \{(R, \rho(R)) \mid R \in \mathcal{E}\}; D' \rangle$.

Im Falle $|\mathcal{E}| = 2$ wird $\langle D; \{(R, R'), (R(D) \setminus R, R(D') \setminus R)\}; D' \rangle$ kürzer notiert:

$$\langle\langle D; R, R'; D' \rangle\rangle$$

Die Definition von $\langle D; \rho; D' \rangle$ fordert, dass die Komponente $D[R]$ ohne Unterschied gegen $D'[\rho(R)]$ austauschbar ist. Mit $\langle\langle D; R, R'; D' \rangle\rangle$ drücken wir also speziell aus, dass die Interaktion der Rolle R mit seiner Umwelt nicht von R' unterschieden werden kann.

Wir betrachten dazu das Dienstnetz aus Abbildung 2.1. Das Dienstnetz PC_3 aus Abbildung 2.7 eine Verfeinerung Rolle $\{Producer\}$ durch die Rolle $R_1 = \{Producer_1, Producer_2\}$.

Analog erkennt man, dass das Dienstnetz PC_2 aus Abbildung 2.8 eine Verfeinerung des Producer/Consumer-Dienstnetzes PC aus Abbildung 2.1 ist. Die Rollen $R_2 = \{Consumer_1, Consumer_2, DecisionMaker\}$ des Dienstnetzes PC_2 verfeinern das Verhalten der Rolle *Consumer* im Dienstnetz PC, d.h. es gilt

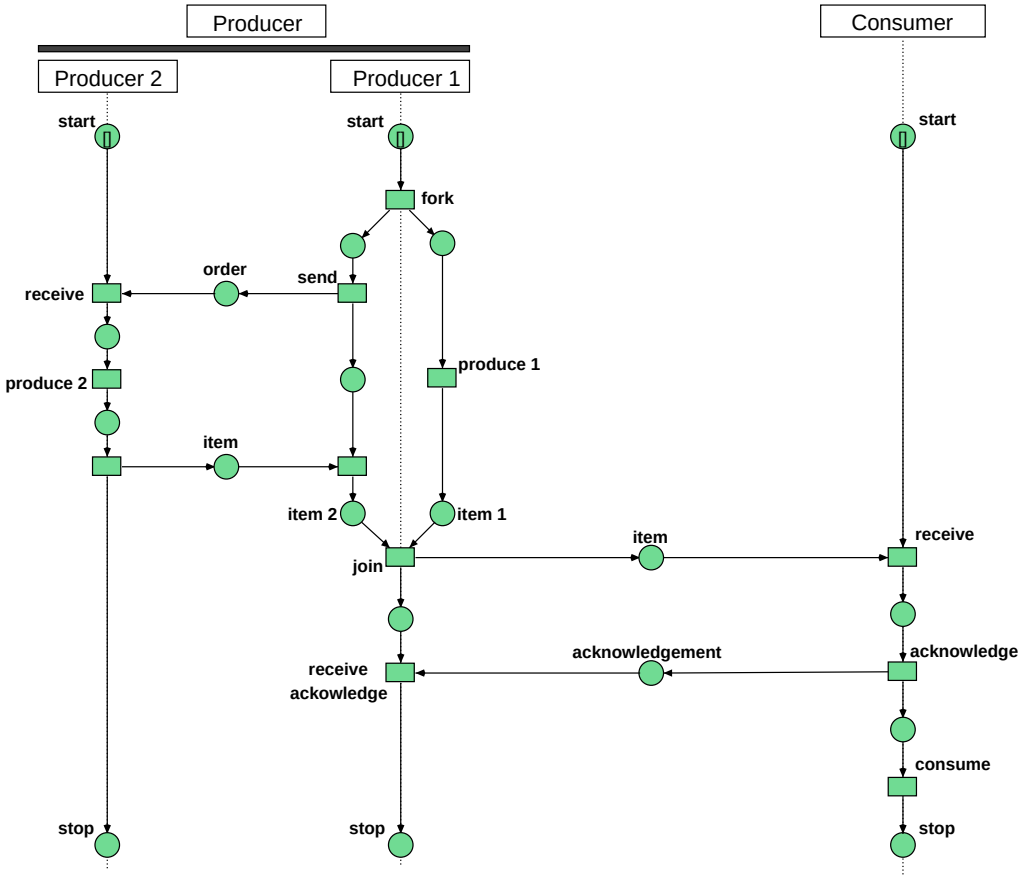


Abbildung 2.7: PC_3: Verfeinerung der Rolle *Producer*

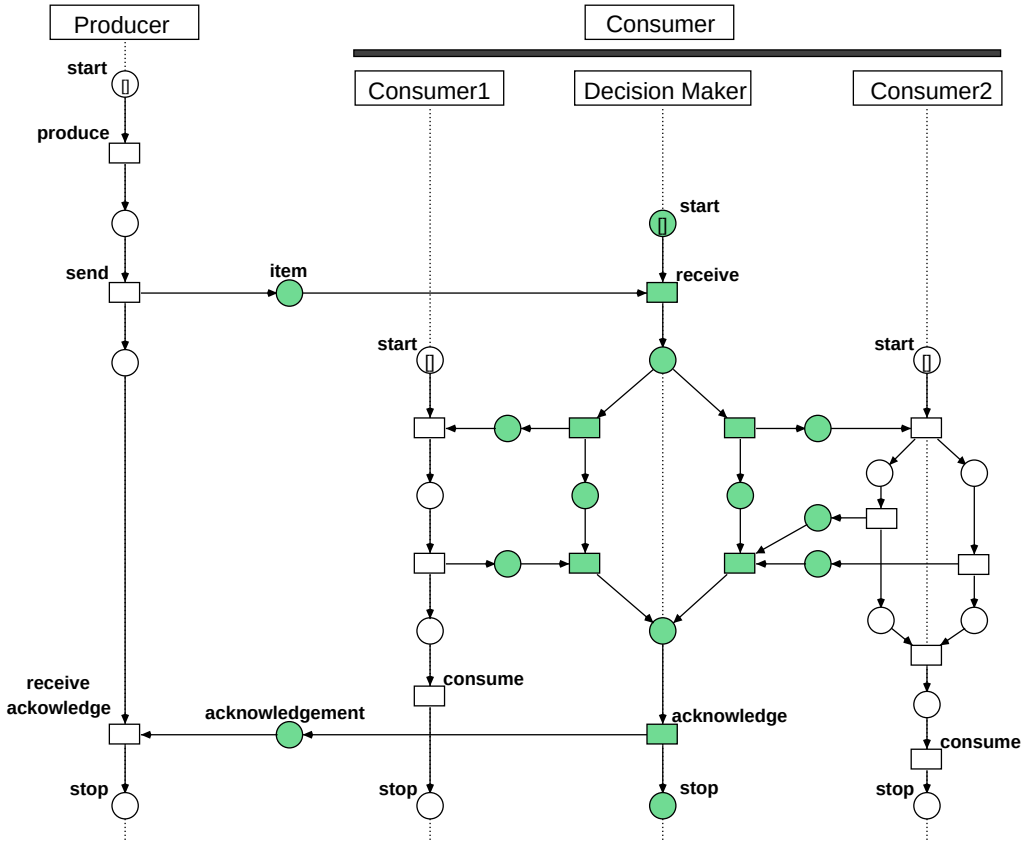


Abbildung 2.8: PC_2: Verfeinerung der Rolle *Consumer*

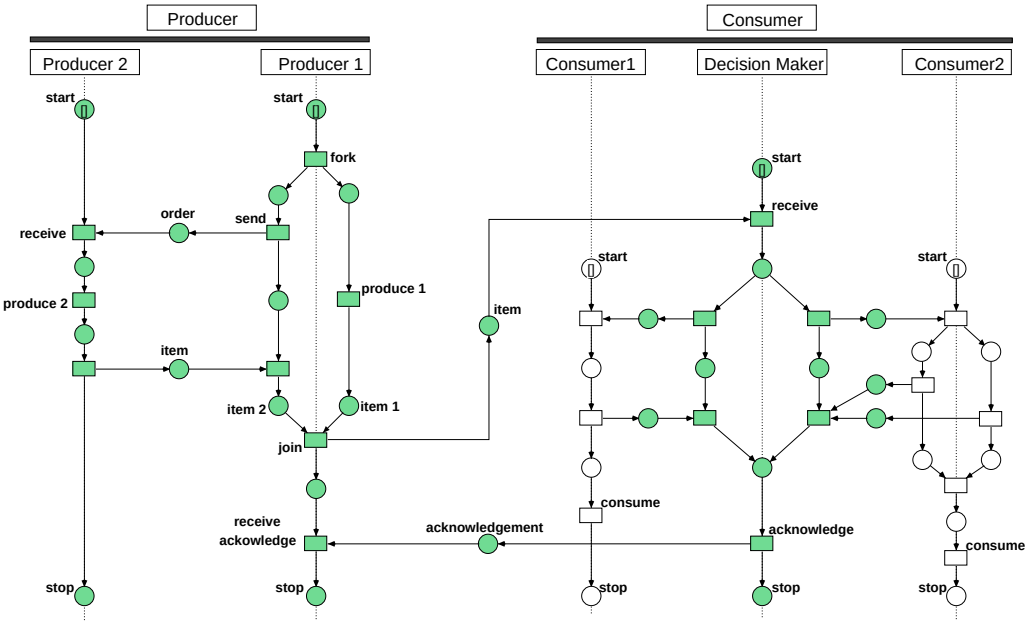


Abbildung 2.9: Die Komposition $PC_4 = PC_3[R_1]||PC_2[R_2]$

$\langle\langle PC; \{Consumer\}, R_2; PC_2 \rangle\rangle$. Eine Kompositionen der beiden Teilverfeinerungen ist in Abbildung 2.9 dargestellt.

Man beachte, dass das der Rolle $Consumer_1$ zugeordnete Teilnetz zwar isomorph zu der Komponente der Rolle *Consumer*, aber nicht mit identisch ist. Dies kann auch nicht sein, denn die Rolle R_2 interagiert bereits extern über Kommunikationskanäle (hier: item und acknowledgement) mit der Rolle *Producer* und diese sind unterschiedlich zu denen, die $Consumer_1$ nutzt.

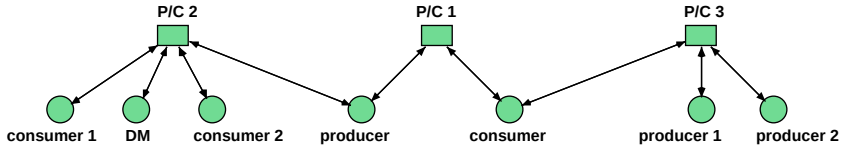


Abbildung 2.10: Übersicht über die Dienst/Rollen-Beziehungen

Eine Übersicht über die Dienst/Rollen-Beziehungen in den Dienstnetzen PC_1, PC_2 und PC_3 ist in Abbildung 2.10 dargestellt. Die Transitionen modellieren die Dienstnetze, die Stellen die beteiligten Rollen und die Kanten die Existenz von sendenden und empfangenden Kommunikationskanälen.

Die Rollenäquivalenz ist eine parametrisierte Äquivalenzrelation.

Theorem 17 *Seien D_1, D_2 und D_3 Dienstnetze. Sei $\mathcal{E}_1 = \{A_1, \ldots, A_n\}$ eine Mengenpartition auf $R(D_1)$, $\mathcal{E}_2 = \{B_1, \ldots, B_n\}$ eine auf $R(D_2)$ und $\mathcal{E}_3 = \{C_1, \ldots, C_n\}$ eine auf $R(D_3)$.*

- *Die Relation ist reflexiv:*
$$\langle D_1; id_{\mathcal{E}_1}; D_1 \rangle$$
- *Die Relation ist symmetrisch:*
$$\langle D_1; \rho; D_2 \rangle \iff \langle D_2; \rho^{-1}; D_1 \rangle$$
- *Die Relation ist transitiv:*
$$\big(\langle D_1; \rho; D_2 \rangle \wedge \langle D_2; \tau; D_3 \rangle\big) \Longrightarrow \langle D_1; (\tau \circ \rho); D_3 \rangle$$

Beweis: Direkt aus Definition 25. q.e.d.

Verhaltensäquivalente Rollenkomponenten können gegeneinander ausgetauscht werden.

Theorem 18 *Die Relation $\langle\langle \cdot; \cdot; \cdot \rangle\rangle$ ist substitutiv:*

$$\langle\langle D_1; A, (B \uplus B'); (D_2[B] \| D_2[\overline{B}]) \rangle\rangle \wedge \langle\langle D_2; B', C; D_3 \rangle\rangle \\ \Longrightarrow \langle\langle D_1; A, (B \cup C); (D_2[\overline{B}] \| D_3[C]) \rangle\rangle$$

Beweis: Für die Substitution ist $N[R_1 \cup R_2] = N[R_1] \| N[R_2]$ auszunutzen: $D_1[A] \approx D_2[B \cup B'] = D_2[B] \| D_2[B']$ und $D_2[B'] \approx D_3[C]$. q.e.d.

Wir können die Rollenaufteilung auch vergröbern, indem wir Rollen zusammenfassen. Nach dem folgenden Kontraktionslemma simulieren dann auch die veröberten Rollenkomponenten einander.

Theorem 19 *Wenn* $\langle D_1; \{(A_1, B_1), (A_2, B_2), \ldots, (A_n, B_n)\}; D_2\rangle$ *für* $n \geq 2$ *gilt, dann gilt auch*

$$\langle D_1; \{(A_1 \cup A_2, B_1 \cup B_2), (A_3, B_3), \ldots, (A_n, B_n)\}; D_2\rangle.$$

Beweis: Sei die Rollenpartition $\{A_1, \ldots, A_n\}$ und $\{B_1, \ldots, B_n\}$ gegeben sowie die Vergröberungen $\{(A_1 \cup A_2), \ldots, A_n\}$ und $\{(B_1 \cup B_2), \ldots, B_n\}$ gegeben.

Wir betrachten nun eine Kommunikationsstelle p in D_1. Die Stelle p verbindet unterschiedliche Rollen: $R({}^\bullet p) = A_i \neq R(p^\bullet) = A_j$. Diese Stelle verbindet auch in D_2 verschiedene Rollen: B_i und B_j, denn nach Definition 24 besitzen beide die gleichen Mengen an Kommunikationsstellen.

Die Stelle p wird in beiden Netzen durch die Kommunikationshandlung t_p, die die Marke von p^{in} nach p^{out} transportiert, verfeinert. Das Schalten von t_p in D_1 kann von D_2 simuliert werden – und umgekehrt.

Gilt $R({}^\bullet p) = A_i$ und $R(p^\bullet) = A_j$ in D_1, dann gilt aufgrund der Substitutionseigenschaft in Definition 25 auch $R({}^\bullet p) = B_i$ und $R(p^\bullet) = B_j$ in D_2. Wir unterscheiden folgende Fälle:

1. Gilt $R({}^\bullet p) = A_1$ und $R(p^\bullet) = A_2$ (oder umgekehrt) in D_1, dann ist $R({}^\bullet p) = B_1$ und $R(p^\bullet) = B_2$ in D_2 und in den beiden Rollenkomponenten $D_1[A_1 \cup A_2]$ und $D_2[B_1 \cup B_2]$ wird p zu einer internen Stelle, so dass es für die Rollenpaarung $\{(A_1 \cup A_2, B_1 \cup B_2), (A_3, B_3), \ldots, (A_n, B_n)\}$ keine zu simulierende Transition t_p gibt. Es ist also nichts zu zeigen.

2. Ist weder $R({}^\bullet p)$ noch $R(p^\bullet)$ gleich A_1 oder A_2 in D_1, dann sind auch B_1 und B_2 in D_2 nicht involviert. Die Simulation in der Vergröberung ergibt sich dann aus der ursprünglichen Simulationseigenschaft.

3. Es gilt $R({}^\bullet p) = A_1$ und $R(p^\bullet) \neq A_2$ (analog für A_2) in D_1. Dann kommuniziert die Rollenkomponente $D_1[A_1 \cup A_2]$ durch t_p mit einer der Rollenkomponente $D_1[A_j]$ mit $j \geq 3$. Dann kommuniziert aber auch $D_2[B_1 \cup B_2]$ durch t_p mit $D_2[B_j]$ und die Aktionen simulieren einander, da sie es auch in der ursprünglichen Form taten.

Die Betrachtung für aus der Perspektive von D_2 verläuft symmetrisch. q.e.d.

2.4 Formale Analyse und Steuerung von Diensten

Zur Spezifikation des Verhaltens hat sich die Verwendung temporaler Logik etabliert. Temporallogiken sind spezielle Modallogiken (Hughes und Cresswell, 1984). Die Gültigkeit temporallogischer Aussagen wird zumeist durch eine *Zustandsraumanalyse* bewiesen. Die als Zustandsraumanalyse (engl. model checking) bekannte Technik geht auf die unabhängig voneinander entwickelten Grundlagen von Emerson und Clarke (1982) sowie Queille und Sifakis (1981) zurück.[4] Die von Emerson und Clarke vorgestellte *Computation Tree Logic* (CTL) wurde in vielen weiteren Arbeiten in der Ausdruckmächtigkeit erweitert. Beispiele sind CTL* (Emerson, 1990) und ACTL (Grumberg und Long, 1994). Weitere Ansätze stellen das μ-Kalkül nach Park (1976) und die Hennessy-Milner-Logik (Hennessy und Milner, 1980) dar. Die

[4] Eine Einführung zu Ansätzen im Bereich der Zustandsraumanalyse finden sich z.B. bei Shankar (1998).

Linear Time Logic (LTL) nach Gabbay u. a. (1980) stellt einen weiteren logikbasierten Ansatz dar, die die Grundlage für einen prominenten Ansatz zur Verifikation reaktiver Systeme nach Manna und Pnueli (1995) darstellt. Zentrale Stärke der LTL gegenüber der CTL ist die Behandlung von starken Fairness-Eigenschaften. Verwandte Arbeiten sind auch das Unity-System nach Chandy und Misra (1989) und die Arbeit von Lamport (1977), die in die *temporal logics of actions* (TLA) (Lamport, 1994) mündete.

2.4.1 Temporallogik für sequentielle Modelle

Die Klasse der zustandsbasierten Logiken ist sehr verbreitet, da sie direkt auf den Modallogiken (vgl. Hughes und Cresswell, 1984) aufbauen können. Das Standardmodell der modalen Logik ist die Kripke-Struktur (W, R), die aus einer Menge an Welten W und deren Verbindungen, dargestellt durch die Relation $R \subseteq W^2$, besteht. Zustandsaussagen lassen sich direkt als Abbildungen von der Menge der Welten in die Menge der Zustände definieren. Die hohe mathematische Formalisierung der Modallogiken begründet die starke Verbreitung der zustandsbasierten Temporallogiken. Zustandsbasierte Temporallogiken lassen sich in zwei Klassen aufteilen: Temporallogiken auf der Basis einer linearen Zeit – wie beispielsweise LTL – sowie auf Basis einer vorwärts verzweigenden Zeit, wie die Logik CTL*. Der Reiz dieser Logiken liegt in ihrer algorithmischen Fundierung. Siehe dazu die Arbeit von Emerson und Clarke (1982) für CTL und die Darstellung nach Pnueli und Lichtenstein (1985) für LTL. Für den Zusammenhang von temporaler Logik zu ω-regulären Sprachen sowie der Automatentheorie siehe Vardi und Wolper (1994) oder Thomas (1990).

Definition 26 *Die Menge der* CTL* *-Formeln CTL$^*(X)$ über einer Menge atomarer Aussagen X ist die Menge aller Pfadformeln:*

1. *Jede atomare Aussage $x \in X$ ist eine Zustandsformel.*
2. *Sind p und q Zustandsformeln, so auch $\neg p$, $(p \vee q)$ und $\mathbf{E}p$.*
3. *Jede Zustandsformel ist eine Pfadformel*
4. *Sind P, Q Pfadformeln, dann auch $\neg P$, $(P \vee Q)$, $\circ P$ und $(P \mathbf{U} Q)$.*

Weitere Operatoren sind von den bestehenden abgeleitet: Wir definieren $\textsc{True} := (x \vee \neg x)$ für ein beliebiges Atom und $\textsc{False} := \neg\textsc{True}$. Der *eventually*-Operator $\Diamond$ ist ebenfalls abgeleitet:

$$\Diamond\psi \quad := \quad (\textsc{True}\ \mathbf{U}\ \psi)$$

Der „leads to“-Operator ist definiert als $\psi \rightsquigarrow \phi := \Box(\psi \Longrightarrow \Diamond\phi)$.

Die bestehenden Operatoren besitzen duale Operatoren:

$$\begin{aligned}
\neg(\phi \wedge \psi) &:= (\neg\phi \vee \neg\psi) \\
\neg\Box\phi &:= \Diamond\neg\phi \\
\neg(\phi\ \mathbf{U}\ \psi) &:= (\neg\psi\ \mathbf{R}\ \neg\phi) \\
\neg\mathbf{A}\psi &:= \mathbf{E}\neg\psi \\
\neg\circ\psi &= \circ\neg\psi
\end{aligned}$$

Die Logik CTL (Emerson und Clarke, 1982) ist eine Einschränkung von CTL*, derart, dass eine Zustandsmodalität nur in Kombination mit einer Pfadmodalität erlaubt ist, d.h. dass beispielsweise **E** nur zusammen mit **U** auftreten darf. Durch diese Einschränkung besitzt CTL ein Fixpunkt-Charakterisierung, was die Grundlage einer effizienten Prozedur für die Zustandsraumanalyse bildet (siehe McMillan, 1993).

Hauptvertreter der Temporallogiken auf der Basis einer linearen Zeit ist die Linear Time Logic (LTL) nach Gabbay u. a. (1980) bzw. Manna und Pnueli (1995). Zentrale Stärke der LTL gegenüber CTL ist die Behandlung von starken Fairness-Eigenschaften. Die temporale Logik LTL formalisiert nur Zustandsaussaugen, die in allen möglichen Sequenzen gültig sein müssen. LTL ist eine Teillogik von CTL*, bei der keine Pfadquantoren verwendet werden. LTL kann somit als Einschränkung von CTL* verstanden werden, da LTL über allen verzweigenden Pfaden interpretiert wird, d.h. eine LTL-Formel ψ entspricht der CTL*-Formel $\mathbf{A}\psi$, wobei ψ keine Pfadquantoren enthalten darf.

Das *Kripkemodell* $M = (K, \alpha, z_s)$ dieser Logik besteht aus einer Kripke-Struktur $K = (Z, \rightarrow)$ zusammen mit einer Bewertungsfunktion $\alpha : Z \rightarrow 2^X$, die jedem Zustand die Menge der gültigen Atome zuweist, und dem Initialzustand $z_s \in Z$. Die Welten der Kripkestruktur werden im Kontext der Temporallogik als Zustände und die Relation zwischen Welten als Zustandsübergänge interpretiert. Die Gültigkeit von Aussagen wird auf Aktionssequenzen von K definiert, wobei die Kripkestrukur als Graph verstanden wird. Um nicht zwischen endlichen und unendlichen Abwicklungen unterscheiden zu müssen, nehmen wir an, dass jeder Zustand mindestens einen Nachfolger hat. Dies ist keine Einschränkung, da jedes System in diese Form bringen können, indem ein neuer Zustand hinzugefügt wird und dieser der Nachfolger aller Zustände wird, die zuvor keinen hatten. Somit können wir uns auf die unendlichen Sequenzen beschränken. Sei σ eine unendliche Sequenz von Zustandsübergängen:

$$\sigma = z_0 z_1 z_2 \cdots \quad \text{mit} \quad \forall i \in \mathbb{N} : z_i \rightarrow z_{i+1}$$

Der Pfad σ_k ist definiert als der Pfad, der im k-ten Zustand von σ beginnt:

$$\sigma_k = z_k z_{k+1} \cdots$$

Sei $\Sigma(z)$ die Menge aller unendlichen Zustandssequenzen, die mit z beginnen.

Die Gültigkeit von Zustandsaussagen ergibt sich relativ zu einem Zustand z_0. So bedeutet $\mathbf{E}p$, dass von diesem Zustand ein Pfad σ ausgeht, der p erfüllt. Die Gültigkeit von Pfadaussagen wird auf Pfaden interpretiert. Gültigkeit notieren wir als $M, \sigma \models P$. Ist das Modell klar, so notieren wir nur kurz $\sigma \models P$. Die Aussage $\bigcirc P$ beschreibt, dass P im Nachfolgezustand z_1 gilt, $P\ \mathbf{U}\ Q$ (engl. *until*) beschreibt, dass in einem erreichbaren Zustand z_n das Prädikat Q gelten wird und dass in allen Zwischenzuständen z_i die Aussage P gilt.

Definition 27 *Sei $M = (K, \alpha, z_s)$ ein Modell. Die Gültigkeit einer* CTL**-Formel*

P in einem Pfad $\sigma = z_0 z_1 \cdots$ wird mit $M, \sigma \models P$. Sie ist folgendermaßen definiert:

$$
\begin{array}{lcl}
M, z_0 \models x & \Longleftrightarrow & x \in \alpha(z_0) \\
M, z_0 \models \neg p & \Longleftrightarrow & M, z_0 \not\models p \\
M, z_0 \models (p \vee q) & \Longleftrightarrow & M, z_0 \models p \text{ oder } M, z_0 \models q \\
M, z_0 \models \mathbf{E}p & \Longleftrightarrow & \exists \sigma \in \Sigma(z_0) : M, \sigma \models p \\
M, \sigma_k \models p & \Longleftrightarrow & z_k \models p \\
M, \sigma_k \models \neg P & \Longleftrightarrow & M, \sigma_k \not\models P \\
M, \sigma_k \models (P \vee Q) & \Longleftrightarrow & M, \sigma_k \models P \text{ oder } \sigma_k \models Q \\
M, \sigma_k \models \bigcirc P & \Longleftrightarrow & M, \sigma_{k+1} \models P \\
M, \sigma_k \models (P \ \mathbf{U} \ Q) & \Longleftrightarrow & \exists n \geq k : M, \sigma_n \models Q \wedge \forall k \leq i < n : M, \sigma_i \models P
\end{array}
$$

Eine Aussage P ist gültig, notiert $M \models P$, wenn sie in z_0 gültig ist.

Die Korrektheitsbedingung für Workflownetze ist ein gutes Beispiel für eine Spezifikation mittels Temporallogik. Wollen wir nicht ausschließlich korrekte Workflow-Netzen betrachten, sondern wollen wir auch solche Workflow-Netze zulassen, die nicht notwendigerweise korrekt sind, weil sie in einigen, aber nicht in allen erreichbaren Markierungen terminieren können, dann müssen wir sie steuern, weil sie die Terminierungsbedingungen verletzen.

Dies können wir mit der folgenden Formel beschreiben, die alle Schaltfolgen beschreibt, die ψ erfüllen und gleichzeitig stets die Option zu terminieren besitzen.

$$\forall M \in RS(D, M_0) : \psi \wedge \exists M' \in RS(D, M) : \pi(M') = m_f \tag{2.10}$$

Wählen wir speziell $\psi = \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f$, so beschreibt (2.10) gerade die Bedingungen der Termination und Fortsetzbarkeit, die für die Korrektheit gefordert sind (vgl. Definition 13).

Definieren wir das Atom x_f und seine Belegung als $x_f \in \alpha(M) \iff (\pi(M) = m_f)$, dann können wir diese Eigenschaft in der Temporallogik CTL$^*(X)$ formulieren:

$$\mathbf{A}\Box(\psi \wedge \mathbf{E}\Diamond x_f) \tag{2.11}$$

2.4.2 Temporallogik für partiell geordnete Modelle

Wir haben bislang Temporallogiken betrachtet, die auf dem Modell total geordneter Ereignisse basiert. In einem Petrinetz-Prozess ist die Nebenläufigkeit unmittelbar zu erkennen, denn jeder Stellen-Schnitt C aktiviert im Allgemeinen eine Menge U an Transitionen, die dann alle nebenläufig zueinander sind, d.h. in einer Sequentialiserung könnte jede Transition $e \in U$ als „erste“ vorkommen. Aufgrund dieser direkten Repräsentation sind die ersten Ansätze für eine temporale Logik partiell geordneter Zustände auf Basis von Petrinetz-Prozessen definiert. Zentrale Arbeiten zu diesem Komplex stammen von Reisig (1988), von Esparza (1994) und von Kindler (1995), die die topologische Unterteilung von Sicherheits- und Lebendigkeitseigenschaften von linearen auf partielle Ordnungen übertragen.

Gerade die Berücksichtigung von Nebenläufigkeit erscheint in vielen Ansätzen als natürliche Antwort auf das Problem der „Zustandsraumexplosion“. Beispielsweise besitzt die Komposition n unabhängiger Binärvariablen einen Zustandsraum mit 2^n Zuständen („Explosion des Zustandsraums“). Sei der Initialzustand $(0, \ldots, 0)$. Es existieren $n!$ Sequentialisierungen, die im sequentiellen Modell alle n Variablen nebenläufig von 0 nach 1 ändern. Es ist für dieses Beispiel jedoch gar nicht notwendig, alle diese Sequenzen zu überprüfen, da sie alle den gleichen Prozess beschreiben.

Unabhängigkeit von Zuständen und Ereignissen wird im Bereich des „partial order model checking“ ausgenutzt, um den Suchraum zu verkleinern (siehe dazu Clarke u. a., 1989; Valmari, 1990, 1994; Godefroid, 1996; Larsem und Thomson, 1991; Grumberg und Long, 1994). Halbordnungssemantiken sind ein Ansatz, um ungeordnete Ereignisse direkt als solche zu behandeln, ohne alle ihre Sequentialisierungen zu betrachten. Diese Reduktion resultiert in einer geringeren algorithmischen Komplexität. Eine solche Betrachtung zu partiellen Ordnungen findet sich bei Bradfield (1992), McMillan (1993) und Godefroid (1996).

Die partielle Ordnung macht einen weiteren (im einfachen, sequentiellen Fall überflüssigen) Operator notwendig, der berücksichtigt, dass mehrere Zustände der direkte Nachfolger sein können. (siehe dazu Desel u. a., 1992). Dieser Operator verallgemeinert den Operator $\circ$ und wird durch $\textcircled{\vee}$ notiert. Die Aussage $\textcircled{\vee}\psi$ beschreibt, dass eine Eigenschaft in einem der direkt erreichbaren Folgezustände gilt. Der Operator $\textcircled{\vee}$ besitzt auch einen dualen Operator $\textcircled{\wedge}$, definiert durch $\textcircled{\wedge}\psi := \neg\textcircled{\vee}\neg\psi$, der besagt, dass eine Eigenschaft in allen direkten Folgezuständen gilt. Betrachten wir die Erweiterung von LTL für partielle Ordnungen nach Kindler (1995):

Definition 28 *Die Menge $LTL_{po}(X)$ der LTL-Formeln für partielle Ordnungen über einer Menge atomarer Aussagen X ergibt sich induktiv.*

1. *Jede atomare Aussage $x \in X$ ist eine $LTL_{po}(X)$-Formel.*
2. *Sind P, Q $LTL_{po}(X)$-Formeln, dann auch $\neg P$, $(P \vee Q)$, $\textcircled{\vee}P$ und $(P \mathbf{U} Q)$.*

Diese Logik wird durch die durch Prozesse eines Petrinetzes interpretiert. Das *Prozessmodell* $M = (Proc(N), \alpha)$ für $LTL_{po}(X)$ besteht aus der Menge aller Prozesse π eines Netzes N und einer Bewertungsfunktion $\alpha = (\alpha_\pi : \mathcal{C}_\pi \to 2^X \mid \pi \in Proc(N))$, die jedem Stellenschnitt C des Prozesses π die Menge der gültigen atomaren Aussagen zuweist.

Definition 29 *Sei $M = (Proc(N), \alpha)$ ein Prozessmodell für $LTL_{po}(X)$ und sei $\pi = (K, \phi) \in Proc(N)$ ein Prozess. Die Gültigkeit einer temporalen Aussage P in einem Stellenschnitt $C \in \mathcal{C}_\pi$ ist induktiv definiert:*

$$
\begin{array}{lcl}
\pi, C \models x & \iff & x \in \alpha_\pi(C) \\
\pi, C \models \neg P & \iff & \pi, C \not\models P \\
\pi, C \models (P \vee Q) & \iff & \pi, C \models P \text{ oder } \pi, C \models Q \\
\pi, C \models \textcircled{\vee}P & \iff & \exists C' \in \mathcal{C}_K : C \to C' \wedge \pi, C' \models P \\
\pi, C \models (P \mathbf{U} Q) & \iff & \exists C' \in \mathcal{C}_K : C \xrightarrow{*} C' \wedge \pi, C' \models Q \wedge \\
 & & \forall C'' \in \mathcal{C}_K : C \xrightarrow{*} C'' \xrightarrow{+} C' \Longrightarrow \pi, C'' \models P
\end{array}
$$

Der Prozess $\pi = (K, \phi)$ erfüllt P, wenn P im initialen Schnitt $^\circ K$ gilt:

$$\pi \models P \iff \pi, {}^\circ K \models P$$

Ein Netz N erfüllt eine Aussage, wenn diese in all seinen Prozessen gilt.

$$N \models P \iff \forall \pi \in Proc(N) : \pi \models P$$

Analog zur Interpretation von LTL_{po} auf Prozessen kann auch die Logik CTL auf partiellen Ordnungen interpretiert werden. Dazu betrachtet man an Stelle der

Menge aller Prozesse das Prozessentfaltungsnetz, das – analog zum Abwicklungsbaum für sequentielle Ordnungen – alle Verzweigungsmöglichkeiten enthält (siehe dazu Esparza, 1994).

Wir bezeichnen die Menge aller Prozesse π eines Netzes N, die eine Eigenschaft ψ erfüllen, mit:

$$Proc(N, \psi) = \{\pi \in Proc(N) \mid \pi \models \psi\} \tag{2.12}$$

Analoge Notationen verwenden wir für Dienste D und Rollenkomponenten $N[R]$ bzw. $D[R]$.

Wenn wir speziell die Prozesse von Rollenkomponenten $D[R]$ betrachten, so erhalten wir als Verallgemeinerung von Theorem 11 das folgende Kompositionstheorem.

Theorem 20 *Sei $R_1, R_2 \in \mathcal{R}$ mit $R_1 \cap R_2 = \emptyset$.*

$$Proc(D[R_1], \psi_1) \cap Proc(D[R_2], \psi_2) = Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$$

Beweis: Sei $K \in Proc(D[R_1], \psi_1) \cap Proc(D[R_2], \psi_2)$, dann gilt $K \models \psi_1$ und $K \models \psi_2$ und damit auch $K \models (\psi_1 \wedge \psi_2)$. Da $Proc(\Gamma, \psi) \subseteq Proc(\Gamma)$ gilt, folgt $K \in Proc(D[R_i])$ für $i = 1, 2$ und mit Theorem 11 auch $K \in Proc(D[R_1] \| R[R_2])$. Insgesamt also $K \in Proc(D[R_1] \| R[R_2], \psi_1 \wedge \psi_2)$.

Sei $K \in Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$, dann gilt $K \models (\psi_1 \wedge \psi_2)$ und damit $K \models \psi_1$ und $K \models \psi_2$. Wenn $K \in Proc(D[R_1] \| D[R_2], \psi_1 \wedge \psi_2)$, dann auch $K \in Proc(D[R_1] \| D[R_2]$ und mit Theorem 11 auch $K \in Proc(D[R_1]) \cap Proc(R[R_2])$. Insgesamt also $K \in Proc(D[R_i], \psi_i)$ für $i = 1, 2$. q.e.d.

2.4.3 Gesteuerte Dienste

Wir sind nun daran interessiert, ob ein Dienst korrekt bleibt, wenn wir sein Verhalten relativ zu einer Eigenschaft ψ betrachten. Wenn D ein Dienst ist und ψ eine Formel, so bezeichnen wir mit $(D \times \psi)$ den durch ψ gesteuerten Dienst.

Schränken wir die Erreichbarkeitsmenge $RS(D, M_0)$ eines Dienstnetz D auf solche Zustände M ein, die $\psi \in \mathrm{CTL}^*(X)$ erfüllen, so erhalten wir die Menge $RS(D \times \psi, M_0) := \bigcup_{i \in \mathbb{N}} RS_i(D \times \psi, M_0)$ mit:

$$\begin{aligned} RS_{k+1}(D \times \psi, M_0) &:= \{M' \mid \exists M \in RS_k(D \times \psi, M_0) : M \xrightarrow{t,\alpha} M' \wedge M' \models \psi\} \\ RS_0(D \times \psi, M_0) &:= \begin{cases} \{M_0\}, & \text{falls } M_0 \models \psi \\ \emptyset, & \text{sonst} \end{cases} \end{aligned}$$

Die Menge $RS(D \times \psi, M_0)$ ist also genau dann leer, wenn M_0 die Eigenschaft ψ nicht besitzt.

Wir schränken den Erreichbarkeitsgraph $RG(D, M_0)$ eines Dienstnetz D auf solche Zustände M ein, die ψ erfüllen, indem wir die Knoten von $RG(D, M_0)$ auf $RS(D \times \psi, M_0)$ einschränken.

Die Korrektheit eines gesteuerten Dienstnetz $D \times \psi$ wird analog zu der des Dienstes D formuliert (vgl. Definition 13), nur dass jeweil $RS(D, M_0)$ durch $RS(D \times \psi, M_0)$ ersetzt werden muss.

Definition 30 *Wenn D ein Dienst ist und $\psi \in \mathrm{CTL}^*(X)$ eine Formel, so bezeichnen wir mit $(D \times \psi)$ den durch ψ gesteuerten Dienst. Ein gesteuertes Dienstnetz*

$(D \times \psi)$ *heißt* korrekt*, wenn gilt:*

$$\begin{aligned}&\forall M \in RS(D \times \psi, M_0) : \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f \\ &\wedge \forall M \in RS(D \times \psi, M_0) : \exists M' \in RS(D \times \psi, M) : \pi(M') = m_f \\ &\wedge \forall t \in T : \exists M_0 : f(M_0) = m_0 \wedge \exists M \in RS(D \times \psi, M_0) : M \xrightarrow{t}\end{aligned}$$

Ein gesteuertes Dienstnetz $(D \times \psi)$ heißt schwach korrekt*, wenn nur die ersten beiden Bedingungen gelten*

Ein gesteuerte Dienstnetz $D \times \psi$ heißt strukturell (schwach) korrekt*, wenn es für alle Anfangsmarkierungen M mit $\pi(M) = m_0$ (schwach) korrekt ist.*

Theorem 21 *Sei D ein Dienst, der in M_0 terminieren kann, d.h. $\exists M \in RS(D, M_0) : \pi(M) = m_f$, dann existiert eine Steuerung ψ, so dass $D \times \psi$ schwach korrekt ist.*

Beweis: D kann terminieren. Also existiert eine Schaltfolge $M_0 \xrightarrow{t_1,\alpha_1} M_1 \xrightarrow{t_2,\alpha_2} \cdots M_n = M$ mit $\pi(M) = m_f$. Wählen wir speziell

$$\psi = \bigwedge_{k=0}^{n} \circ^k (M = M_k) = \Big((M = M_0) \wedge \circ (M = M_1) \wedge \circ\circ (M = M_2) \wedge \cdots \Big),$$

so erlaubt die Steuerung nur diese Schaltfolge, was zeigt, dass der Dienst $D \times \psi$ korrekt ist. q.e.d.

2.5 Dienstklassen

Dienstklassen sind Mengen von Dienstnetzen, die konsistent bezüglich der verwendeten Rollen sind. Formulieren wir im folgenden Konsistenz. Die Rollen der Dienstnetze einer Dienstklasse sind nicht notwendigerweise disjunkt. Es ist möglich, dass die gleiche Rolle r in zwei Dienstnetzen D_1 und D_2 vorkommt. In diesem Fall fordern wir aber, dass die Rolle in beiden Netzen identische Prozesse beschreiben:

$$\langle\langle D_1; R, R; D_2 \rangle\rangle$$

Dadurch wird ausgedrückt, dass Interaktion und Rollen zwei untrennbare Konzepte sind, d.h. dass unter einer abstrakten Betrachtungsweise die Interaktion mit dem Verhältnis der an ihr beteiligten Rollen gleichzusetzen ist.

Zu einer Menge an Diensten $\mathcal{D}$ bezeichnet $\mathcal{D}(R)$ die Menge aller Dienste, die die Rolle $R \in \mathcal{R}$ enthalten:

$$\mathcal{D}(R) = \{D \in \mathcal{D} \mid R(D) \supseteq R\} \tag{2.13}$$

Wir definieren eine Auswahlfunktion $D^{\mathcal{D}}$, die zu jeder Rolle R einen Repräsentanten aus der Menge $\mathcal{D}(R)$ auswählt. Diesen Repräsentanten $D^{\mathcal{D}}(R)$ bezeichnen wir als den *Referenzdienstnetz* von R. Für den Referenzdienst von R gilt offensichtlich:

$$R \subseteq R(D^{\mathcal{D}}(R))$$

Definition 31 *Sei $\mathcal{R}$ eine Rollenstruktur. Eine* Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ *besteht aus einer Menge an Dienstnetzen $\mathcal{D}$ mit*

$$\forall R \in \mathcal{R} : \forall D_1, D_2 \in \mathcal{D}(R) : \langle\langle D_1; R, R; D_2 \rangle\rangle$$

und einer Abbildung $D^{\mathcal{D}}(R) : \mathcal{R} \to \mathcal{D}$ mit

$$\forall R \in \mathcal{R} : D^{\mathcal{D}}(R) \in \mathcal{D}(R)$$

Bemerkung: Mit dem Kontraktionslemma 19 reicht es aus, für Dienstklassen die Äquivalenz $\langle\langle D_1; R, R; D_2\rangle\rangle$ für elementare Rollen $R = \{r\}$ zu fordern, da alle anderen Rollen durch Vereinigung gebildet werden können.

Für jede Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ generiert jede Rolle R die Klasse $\mathcal{D}(R)$, deren Elemente $D \in \mathcal{D}(R)$ sich nicht in ihrem Verhaltem bezüglich der Rolle R unterscheiden lassen.

Für eine Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ definieren wir die Menge aller Dienstkomponentenprozesse:

$$Proc(\mathcal{D}, \mathcal{R}) := \bigcup_{D \in \mathcal{D}} \bigcup_{R \in \mathcal{R}} Proc(D[R]) \tag{2.14}$$

Wir definieren im folgenden die Menge der Formeln für eine Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$. Sei $D \in \mathcal{D}$ ein Dienstnetz. Die Atome sind von der Form $A_{k,q}$, wobei $k \in \mathbb{Q}^{|P_{KK}(D)|}$ und $q \in \mathbb{Q}$ ist, oder von der Form $A_{p,c,n}$, wobei $p \in P_{KK}(D)$, $c \in [\![d(p)]\!]$ und $n \in \mathbb{N}$ ist. Die Menge aller dieser Atome von D wird mit $\mathcal{A}_D$ bezeichnet. Die Menge aller Atome ist $\mathcal{A}_{\mathcal{D}} = \bigcup_{D \in \mathcal{D}} \mathcal{A}_D$.

Zu einer Markierung M ergibt sich die Bewertungsfunktion α als die Familie der α_D, die jeweils folgendermaßen definiert sind:

$$\begin{aligned} \alpha_D(M) \quad = \quad & \{A_{k,c} \mid \textstyle\sum_{p \in P_{KK}(D)} (k(p) \cdot \pi(M)(p)) \leq c\} \cup \\ & \{A_{p,c,n} \mid M(p)(c) \leq n\} \end{aligned}$$

Zu einem Prozess $\pi \in Proc(D)$ erweitert sich dies zu einer Bewertung auf den Schnitten:

$$\alpha_{D,\pi}(C) = \alpha_D(\phi(C))$$

Damit ist dann $PM(\mathcal{D}) = (\bigcup_{D \in \mathcal{D}} Proc(D), \alpha)$ das von D erzeugte Standardprozessmodell der Logik $\mathrm{LTL}_{po}(\mathcal{A}_{\mathcal{D}})$.

Um sowohl die Steuerung mittels des Erreichbarkeitgraphen als auch die Erfüllbarkeit im Prozessmodell betrachten zu können, schränken wir die folgenden Betrachtungen auf Formeln ein, die sowohl in $\mathrm{LTL}(X)$ als auch $\mathrm{LTL}_{po}(X)$ sind.

Definition 32 *Sei $(\mathcal{D}, D^{\mathcal{D}})$ eine Dienstklasse. Die Menge der* Prozessformeln *ist definiert als die Menge:*

$$PF_{\mathcal{D}} := \mathrm{LTL}(\mathcal{A}_{\mathcal{D}}) \cap \mathrm{LTL}_{po}(\mathcal{A}_{\mathcal{D}}) \tag{2.15}$$

Für die Steuerung konvertieren wir jede Prozessformel ψ implizit als eine CTL*-Formel, indem wir den Pfadquantor $\mathbf{A}$ hinzufügen, d.h. wir betrachten $\mathbf{A}\psi$ anstelle von ψ.

Zusammenfassung

In diesem Kapitel haben wir die Interaktion verteilter Systeme mit Hilfe von Dienstnetzen formalisiert. Dienstnetze sind gefärbte Petrinetze, die die Grundstruktur eines Ablaufnetzes besitzen und zudem mit einer Konzeptbeschreibung und einer Rollenstruktur versehen sind.

Da Dienstnetze eng mit den Workflownetzen verwandt sind, haben wir die Korrektheit von Dienstnetzen analog definiert und konnten zeigen, dass sich Korrektheit von Ablaufnetzen entscheiden lässt, indem man diese auf Workflows reduziert.

Ein weiterer wichtiger Aspekt war die Erzeugung von Rollenkomponenten. Wir konnten anhand der Rollenstruktur der Dienstnetze Rollenkomponenten definieren. Rollenkomponenten sind Subnetze von Dienstnetzen, deren Netzknoten alle

der gleichen Rolle zugeordnet sind. Rollenkomponenten sind stets Netzkomponenten und erweisen sich als kompositonale Systeme, die stets miteinander wohlgeformt verknüpft werden können. Rollenkomponenten stellen den Dienst dar, der von der Rolle erbracht wird. Sie stellen implizit eine Spezifikation der Rechte und der Pflichten, die mit dieser Rolle verbunden sind, dar.

Um Dienstnetze koordinieren und steuern zu können, haben wir auch noch den Bezug zur temporalen Logik betrachtet. Dies sowohl für sequentielle als auch für partiell geordnete Modelle.

Im folgenden betrachten wir nun, wie die Agenten in Teams aggregiert werden, um die Dienstnetze „mit verteilten Rollen“ auszuführen.

3 Koordinierung in Agentensystemem

In Multiagentensystemen finden sich Strukturen, die nicht einzelnen Agenten zuzuordnen sind, sondern ausschließlich der Interaktion zwischen ihnen dienen, beispielsweise Koordinationsstrukturen. In diesem Kapitel verschieben wir aufbauend auf (Köhler, 2007) den Betrachtungswinkel weg vom Agenten hin zu einer ganzheitlichen Perspektive, in der wir die Interaktionen zwischen den Agenten in den Mittelpunkt stellen.

Betrachten wir beispielshaft eine Produzenten/Konsumenten Beziehung (engl. producer/consumer scenario). Diese Beziehung wird in einem Markt hergestellt. Die Tätigkeit des Handelns involviert zwei Rollen, nämlich die des Produzenten (*producer*) und die des Konsumenten (*consumer*). Im Markt befinden sich m Agenten $A_1, \ldots, A_m$, die alle gleichermaßen in der Lage sind, als Produzent tätig zu werden. Daneben existieren n Agenten $B_1, \ldots, B_n$, die die Rolle des Konsumenten einnehmen können.

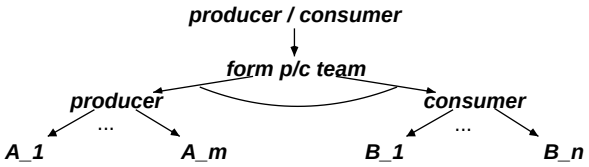


Abbildung 3.1: Producer/Consumer Szenario

Von Interesse ist, wie sich Produzenten und Konsumenten im Markt finden, d.h. wie sich Handelsbeziehungen in der Gestalt von Teamstrukturen formieren. Abbildung 3.1 formalisiert die Beziehungen zwischen den Agenten, den Rollen und der Tätigkeit als AND/OR-Graph: Die Rollen der Tätigkeit sind durch eine konjunktive Kante verbunden, da beide Rollen gleichzeitig benötigt werden. Dagegen sind die Agenten disjunktiv mit den Rollen assoziiert, da jeder Agent eine mögliche Instantiierung darstellt.

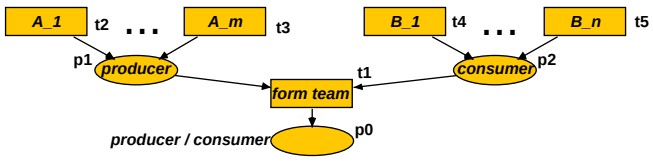


Abbildung 3.2: Formation des Producer/Consumer-Teams

Abbildung 3.1 stellt eine statische Sichtweise auf das System dar. Teamformation ist jedoch ein dynamischer Prozess, bei dem sich die Agenten zu einem Team gruppieren. Um diese Dynamik auszudrücken, modellieren wir das Szenario als Petrinetz, dargestellt in Abb. 3.2. Das Netz, das die Teamformation modelliert, besitzt dabei eine Grundstruktur, die dem AND/OR-Graphen stark ähnelt. Jeder Agent kann die

mit ihm assoziierte Transition schalten, um sich für die Rolle zu bewerben. Ist zu jeder Rolle mindestens eine Bewerbung vorhanden, kann die Transition form team schalten, wodurch die beiden Marken abgezogen werden und eine neue Marke auf der Stelle producer/consumer generiert wird, die das Team darstellt. Man beachte, dass das Modell mit $m + n + 4$ Netzknoten bereits $m \cdot n$ Schaltprozesse besitzt, die jeweils den $m \cdot n$ möglichen Produzent/Konsument Paarungen entsprechen. Das Modell leistet also bereits eine kompaktere Darstellung des Sachverhalts.

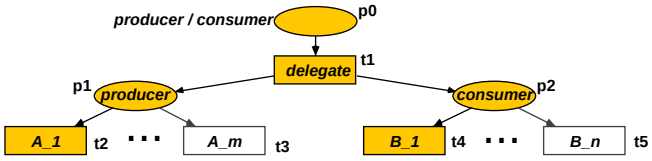


Abbildung 3.3: Bearbeitung des Producer/Consumer-Teams

Das Netz aus Abb. 3.2 zeigt die Sicht der Teamformation. Wir können diesen Prozess genauso aus der Perspektive der Aufgabenbearbeitung betrachten. Die Aufgabe ist es, eine Produzenten/Konsumenten-Beziehung mit Hilfe der im Markt vorhanden Agenten zu realisieren. Dieses Szenario ist in Abb. 3.3 modelliert. Eine Marke auf der producer/consumer aktiviert die Transition delegate (der Korrespondenz zur Transition form team), die die beiden Rollen aktiviert, die wiederum durch Agenten ausgefüllt werden. Der durch die Pfeilrichtung festgelegte Markenfluss beschreibt hierbei die Delegationsbeziehungen. Dies ist nicht mit den Kommunikationsbeziehungen zu verwechseln, denn im allgemeinen kommunizieren alle Rolleninhaber im Nachbereich einer Tätigkeit miteinander. Im Beispiel interagieren also der Produzent und der Konsument – und dies, obwohl sie im gerichteten Graphen nicht durch direkte oder indirekte Delegationsbeziehungen verbunden sind.

Auch dieses Modell ähnelt dem AND/OR-Graphen. Auch die beiden Netzmodelle sind miteinander verwandt: Offensichtlich ist Abb. 3.3 das reverse Netz von Abb. 3.2, d.h. es wurde lediglich die Richtung der Kanten umgekehrt. Das Verhalten der beiden Netze ist trotz ihrer Ähnlichkeiten sehr verschieden, da wir von unterschedlichen Startmarkierungen ausgehen.

Es sollte klar geworden sein, dass die beiden Modelle verwandte Prozesse beschrieben: Die beiden Perspektiven modellieren den gleichen Sachverhalt aus unterschiedlichem Blickwinkel, nämlich zum einen aus dem Blickwinkel der Agenten und zum anderen aus dem der Delegations- und Interaktionsbeziehung. Welchem Blickwinkel wir hier den Vorzug geben ist Geschmackssache. Wir entscheiden uns hier für die Perspektive der Delegationsbeziehung wie sie in Abbildung 3.3 modelliert wurde.

3.1 Aufgabenbearbeitung in Delegationsnetzen

Ein Team lässt sich als ein spezielles Petrinetz (P, T, F) definieren, das wir im folgenden als Delegationsnetz bezeichnen. Teams bearbeiten Aufagen durch Delegation von Teilaufgaben. Jede Aufgabe ist mit einer Rolle identifiziert. Jede *Aufgabe*, wird durch eine Stelle modelliert, jede *Delegation* durch eine Transition. Die Semantik der Kanten ist für (p, t)- und (t, p)-Kanten unterschiedlich: Die Profile $t^\bullet$ im Nachbereich einer Transition t sind genau diejenigen, die t zur Ausführung benötigt (konjunktive Verknüpfung). Jede Transition $t \in p^\bullet$ im Nachbereich von p

beschreibt alternative Wahlmöglichkeiten von Tätigkeiten, die zur Bearbeitung der Teilaufgaben genutzt werden können (disjunktive Verknüpfung).

Da wir die Delegationsperspektive auf Teams gewählt haben, wird jede Tätigkeit durch genau eine Aufgabe angestoßen, wodurch wir fordern können, dass jede Transition t höchstens eine Stelle im Vorbereich besitzt. Es ist zudem wünschenswert, dass der Vorbereich von t nicht leer ist, damit Aufgaben nicht ohne Anlaß generiert werden. Insgesamt ergibt sich dann $\forall t \in T : |{}^\bullet t| = 1$. Die dann eindeutig bestimmte Stelle p im Vorbereich von t bezeichnen wir mit $p(t)$. Außerdem muss $p^\bullet \neq \emptyset$ gelten, damit jede Aufgabe überhaupt bearbeitbar ist.

Definition 33 *Ein* Delegationsnetz *ist ein Petrinetz* $N = (P, T, F)$*, das* $p^\bullet \neq \emptyset$ *für alle* $p \in P$ *und* $|{}^\bullet t| = 1$ *für alle* $t \in T$ *erfüllt.*

Eine Transition mit $t^\bullet \neq \emptyset$ heißt *delegativ*, eine Transition mit $t^\bullet = \emptyset$ heißt *ausführend*.

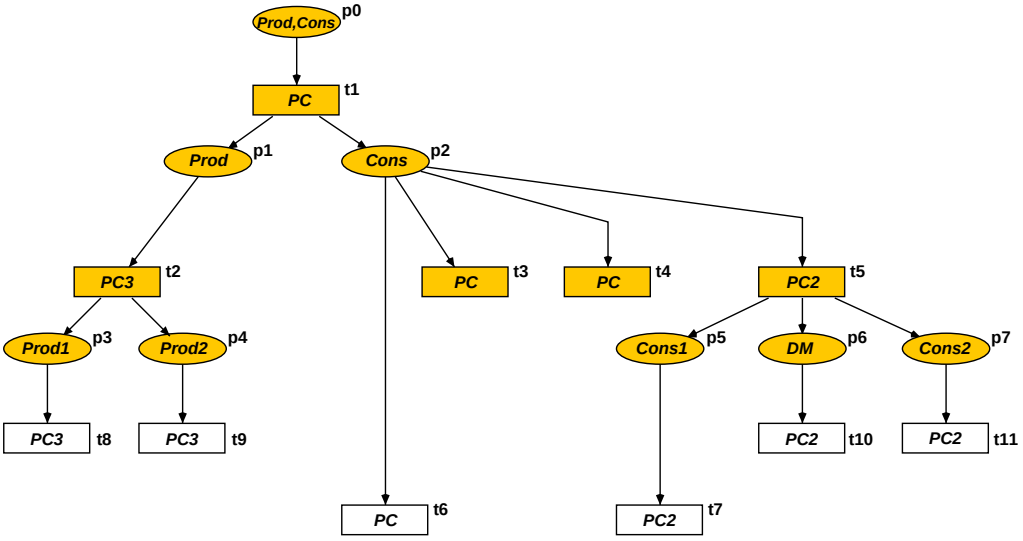


Abbildung 3.4: Ein Delegationsnetz

Sei $N = (P, T, F)$ ein Delegationsnetz. *Tätigkeitspfade* sind Schaltfolgen $w \in T^+$, die das Netz von einer Markierung m in die leere Markierung $\mathbf{0}$ überführen. In dieser Markierung sind alle Teilaufgaben Diensten zugeordnet worden, so dass keinerlei unzugewiesene Aufträge (sprich: Marken) mehr existieren. Für das Netz aus Abbildung 3.4 in der Markierung $m = p_0$ sind dies die Schaltfolgen (modulo Permutation nebenläufiger Ereignisse):

$$t_1t_2t_8t_9t_6, \quad t_1t_2t_8t_9t_3, \quad t_1t_2t_8t_9t_4 \quad \text{und} \quad t_1t_2t_8t_9t_5t_7t_{10}t_{11}$$

Die Menge der Tätigkeitspfade aus einer Markierung m ist:

$$TP(m) := \{w \in T^* \mid m \xrightarrow{w} \mathbf{0}\} \tag{3.1}$$

Wir fragen uns, ob jede Markierung eines R/D-Netzes, die von in der einer initialen erreichbar ist, in die leere Markierung überführt werden kann.

Definition 34 *Sei* $N = (P, T, F)$ *ein P/T-Netz.*

- *Die Markierung m heißt* bearbeitbar*, wenn $\mathbf{0} \in RS(m)$ gilt.*
- *Das Netz N heißt* bearbeitbar*, wenn alle Markierungen m dies sind.*
- *Die Markierung m heißt* sicher bearbeitbar*, wenn für alle $m' \in RS(m)$ auch $\mathbf{0} \in RS(m')$ gilt.*
- *Das Netz N heißt* sicher bearbeitbar*, wenn alle Markierung dies sind.*

Für beliebige Netze ist es sehr aufwendig, die Bearbeitbarkeit zu entscheiden, da das in der Definition befindliche das Erreichbarkeitsproblem im allgemeinen mindestens exponentiell viel Platz benötigt. Es wird sich aber im folgenden zeigen, das wir die Struktureigenschaften der Delegationsnetze nutzen können, um das Problem effizient zu entscheiden. Betrachten wir zunächst die Eigenschaften, die sich aus der Netzstruktur der Delegationsnetze ergeben.

3.1.1 Linearität von Erreichbarkeitsmengen

Wir betrachten die Linearitätseigenschaft von Delegationsnetzen. Wir erweitern die Multimengenaddition + auf Mengen von Multimengen durch:

$$A + B := \{m_1 + m_2 \mid m_1 \in A \wedge m_2 \in B\}$$

Theorem 22 *Sei N ein Delegationsnetz, dann gilt:*

$$RS(m_1 + m_2) = RS(m_1) + RS(m_2)$$

Beweis: Die Inklusion $RS(m_1 + m_2) \supseteq RS(m_1) + RS(m_2)$ folgt aus der Monotonieeigenschaft der Petrinetze.

Nehmen wir an, es gäbe eine Markierung $m \in RS(m_1 + m_2)$, für die $m \notin RS(m_1) + RS(m_2)$ gilt. Dann gäbe es eine Markierung $m' = m'_1 + m'_2$ mit $m'_i = m'_1 \in RS(m_i)$ mit $i = 1, 2$, die eine Transition t aktiviert, die weder in m'_1 noch in m'_2 aktiviert ist, denn gäbe es kein solches m', dann könnte auch kein m erreicht werden. Wegen der Bedingung $|{}^\bullet t| = 1$ kann es für Delegationsnetze eine solche Transition nicht geben. Also gilt $RS(m_1 + m_2) \subseteq RS(m_1) + RS(m_2)$. q.e.d.

Die erreichbaren Markierungen $RS(\sum_{i=1}^{n} p_i)$ sind daher bereits durch die elementaren Mengen $RS(\{p_i\})$ charakterisiert. Analog ergibt sich eine Linearität für die Schaltfolgen, hier für den Shuffle-Operator $\sqcup\!\sqcup$, der Schaltfolgen ineinander mischt.

Theorem 23 *Sei N ein Delegationsnetz, dann gilt:*

$$FS(m_1 + m_2) = FS(m_1) \sqcup\!\sqcup\ FS(m_2)$$

Beweis: Jede Schaltfolge von m_1 kann beliebig mit denen von m_2 gemischt werden, um eine Schaltfolge von $m_1 + m_2$ zu erhalten, denn wegen $|{}^\bullet t| \leq 1$ aktiviert eine Markierungssumme $m_1 + m_2$ keine Transitionen, die nicht bereits in m_1 oder in m_2 aktiviert gewesen sind.

Umgekehrt lässt sich jede Schaltfolge von $m_1 + m_2$ zerlegen, da jede Marke unabhängig von den anderen bearbeitet wird. q.e.d.

3.1.2 Kontextfreiheit der Tätigkeitspfade

Es erscheint naheliegend, die Schaltfolgen eines Delegationsnetzes $N = (P, T, F)$ durch eine kontextfreie Grammatik zu beschreiben. Die Stellen $p \in P$ sind die Nonterminale (Variablen):

$$V_G = \{A_p \mid p \in P\}$$

Die Transitionen $t \in T$ sind die Terminale:

$$T_G = \{a_t \mid t \in T\}$$

Sei $P = \{p_1, \ldots, p_n\}$. Jeder Markierung m wird das folgende Wort zugeordnet:

$$\alpha(m) = A_{p_1}^{m(p_1)} \cdots A_{p_n}^{m(p_n)} \tag{3.2}$$

Jeder Transition t mit $t^\bullet = \{p_1, \ldots, p_n\}$ werden die $n+1$ Produktionen zugewiesen:

$$A_{p(t)} \to a_t A_{p_1} \cdots A_{p_n} \mid A_{p_1} a_t A_{p_2} \cdots A_{p_n} \mid \ldots \mid A_{p_1} \cdots a_t A_{p_n} \mid A_{p_1} \cdots A_{p_n} a_t$$

Hierbeit ist $p(t)$ die eindeutig bestimmte Stelle im Vorbereich von t. Die Startproduktion ist $S \to \alpha(m)$. Die so definierte Grammatik $G(N, m)$ ist offensichtlich in Greibach Normalform.

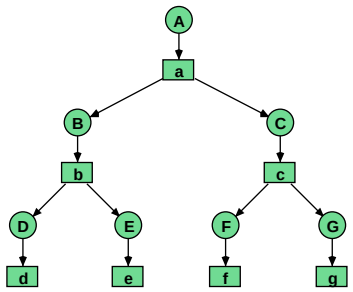


Abbildung 3.5: Ein Delegationsnetz

Mit der so definierten Grammatik $G(N, m)$ erhalten wir aber nur eine Approximation an die Menge der Schaltfolgen, denn die Transitionen gleicher Schalttiefe werden nach Teilbäumen sortiert. So ist für das Delegationsnetz in Abbildung 3.5 die Schaltfolge $w = abdcefg$ möglich, d.h. $w \in FS(N)$. Dieses w ist aber nicht ableitbar:

$$S \Longrightarrow aBC \Longrightarrow abDEC \Longrightarrow abDEcFG$$

Man erkennt, dass in w die Transition c zwischen das d und das e geraten ist, was die Grammatik, die immer jeden Teilbäume komplett entwickelt, nicht leisten kann. Nach der Wahl der Produktion $S \to aBC$ kann kein von B entwickeltes Symbol mehr nach einem von C entwickelten erzeugt werden.

Im allgemeinen gilt also nur $L(G(N, m)) \subseteq FS(N)$. Die Schaltfolgen dieser Bäume sind als *Szillard-Sprachen* bekannt (vgl. Salomaa, 1987). Für unsere Zwecke ist dies aber keine gravierende Einschränkung, denn zu jeder Schaltfolge, die von $G(N, m)$ nicht erzeugt werden kann, existiert ein Schaltwort in $L(G)$, das sich nur bzgl. der Vertauschung nebenläufiger Ereignisse unterscheiden. Dies gilt sogar, wenn wir die Menge der Regeln auf einen Produktionstyp einschränken, bei dem wir die Transition an den Anfang setzen:

$$A_{p(t)} \to a_t A_{p_1} \cdots A_{p_n}$$

Definition 35 *Sei $N = (P, T, F)$ ein Delegationsnetz. Dann ist die kontextfreie Grammatik $G(N, m) = (T_G, V_G, R_G, S)$ definiert als:*

1. *Die Menge der Terminale ist $T_G = \{a_t \mid t \in T\}$.*
2. *Die Menge der Variablen ist $V_G = \{A_p \mid p \in P\} \uplus \{S\}$.*
3. *Die Menge der Regeln ist gegeben als*

$$R_G = \{S \to \alpha(m)\} \cup \{A_{p(t)} \to a_t A_{p_1} \cdots A_{p_n} \mid t \in T \wedge t^\bullet = \{p_1, \ldots, p_n\}\}.$$

4. *Das Startsymbol ist S.*

Erreichbare Nonterminale $A \in V$ einer kontextfreien Grammatik $G = (T, V, R, S)$ sind solche, die sich von dem Startsymbol ableiten lassen: $S \overset{*}{\Longrightarrow} \alpha A \beta$ mit $\alpha, \beta \in (X \cup V)^*$. Die Menge der erreichbaren Nonterminale (engl. reachable variables)

$$RV(G) = \{A \in V \mid \exists \alpha, \beta \in (X \cup V)^* : S \overset{*}{\Longrightarrow} \alpha A \beta\}$$

lassen sich schrittweise berechnen. Definiere dazu:

$$\begin{aligned} RV_0(G) &= \{S\} \\ RV_{n+1}(G) &= RV_n(G) \cup \{B \in V \mid \exists (A \to \alpha B \beta) \in P : A \in RV_n(G)\} \end{aligned}$$

Produktive Nonterminale $A \in V$ einer kontextfreien Grammatik $G = (T, V, R, S)$ sind solche, von denen sich eine Terminalwort $w \in T^*$ ableiten lässt: $A \overset{*}{\Longrightarrow} w$. Die Menge der produktiven Nonterminale (engl. productive variables)

$$PV(G) = \{A \in V \mid A \overset{*}{\Longrightarrow} w \wedge w \in T^*\}$$

lassen sich schrittweise berechnen. Definiere dazu:

$$\begin{aligned} PV_0(G) &= T \\ PV_{n+1}(G) &= PV_n(G) \cup \{A \in V \mid \exists (A \to w) \in P : w \in PV_n(G)^*\} \end{aligned}$$

Die Mengen $PV_1(G), PV_2(G), \ldots$ sind monoton wachsend (bzgl. der Mengeninklusion) und durch die endliche Menge V nach oben beschränkt, so dass $PV_{k+1}(G) = PV_k(G)$ für einen Index $k \leq |V|$ gilt. Für diesen Index gilt dann $PV_k(G) = PV(G)$. Analog für $RV(G)$.

Theorem 24 *Sei $N = (P, T, F)$ ein Delegationsnetz.*

1. *Die Markierung m ist genau dann bearbeitbar, wenn alle Nonterminale A_p der kontextfreien Grammatik $G(N, m)$ mit $m(p) > 0$ produktiv sind.*
2. *Die Markierung m ist genau dann sicher bearbeitbar, wenn alle erreichbaren Variablen A_p der kontextfreien Grammatik $G(N, m)$ mit $m(p) > 0$ produktiv sind.*

Beweis:

1. Wenn ein Nonterminal $A_p \in N_G$ produktiv ist, dann gilt $A_p \stackrel{*}{\Longrightarrow} w'$ Da $S \rightarrow \alpha(m)$ gilt und $\alpha(m)$ alle A_p mit $m(p) > 0$ enthält, haben wir $S \stackrel{*}{\Longrightarrow} w$ mit $w = a_{t_{i_1}} \cdots a_{t_{i_n}} \in T_G^*$. Nach Konstruktion von $G(N, m)$ ist $T_G = \{a_t \mid t \in T\}$. Da die Produktionen der Grammatik dem Schaltverhalten entsprechen, ist $t_{i_1} \cdots t_{i_n}$ ein Tätigkeitspfad. Also ist $\mathbf{0}$ von m in N erreichbar.

   Ist umgekehrt $\mathbf{0}$ erreichbar, dann gibt es eine Schaltfolge $w \in T^*$ mit $m \stackrel{w}{\rightarrow} \mathbf{0}$. Also gilt $S \Longrightarrow \alpha(m) \stackrel{*}{\Longrightarrow} w$ und damit gibt es eine Ableitung von A_p, in der alle Nonterminale, die ja den Stellen entsprechen, verschwinden, d.h. die A_p sind produktiv.

2. Ist m' erreichbar von m und $\mathbf{0}$ von m', dann gilt $S \Longrightarrow \alpha(m) \stackrel{*}{\Longrightarrow} \alpha(m')$. Dies gilt genau dann, wenn alle A_p with $p \in \{p \mid m'(p) > 0\}$ produktiv sind und alle A_{p_i} mit $\alpha(m') = A_{p_1}^{m'(p_1)} \cdots A_{p_n}^{m'(p_n)}$ erreichbare Variablen in $G(N, m)$ sind.

q.e.d.

Beschränktheit entspricht den Zyklen im Netz, bzw. den Zyklen der Grammatik. Ein Variable A kann sich selbst generieren, wenn $\alpha, \beta \in (X \cup V)^*$ existieren, so dass $A \stackrel{*}{\Longrightarrow} \alpha A \beta$ gilt. Gilt $|\alpha\beta| > 0$, dann generiert sich A ansteigend.

Lemma 1 *Ein Delegationsnetz (N, m_0) ist genau dann unbeschränkt, wenn in $G(N, m_0)$ eine erreichbare Variable A existiert, die sich selbst ansteigend generieren kann.*

Beweis: Ein Netz ist genau dann unbeschränkt, wenn es überdeckende Schaltfolge hat:

$$m_0 \stackrel{*}{\rightarrow} m_1 \stackrel{*}{\rightarrow} m_2 \quad \text{und} \quad m_1 < m_2$$

Dies ist für Delegationsnetze genau dann der Fall, wenn in $G(N, m_0)$ auch $S \stackrel{*}{\Longrightarrow} \alpha_1 A \alpha_2$ und $A \stackrel{*}{\Longrightarrow} \beta_1 A \beta_2$. Außerdem muss $m_1 < m_2$ und damit auch $|\beta_1\beta_2| > 0$ glten. Also ist A eine erreichbare Variable ist, die sich selbst ansteigend generieren kann. q.e.d.

3.2 Rollen/Dienst-Netze und Teams

Wir erweitern nun Delegatiosnetze um Anschriften, die die Rollen und Dienste näher charakterisieren. Diese Netze bezeichnen wir als Rollen/Dienst-Netze (kurz: R/D-Netze). Für R/D-Netze wird jedem $p \in P$ durch $R : P \rightarrow \mathcal{R}$ seine Rolle zugewiesen (vgl. Def. 9). Außerdem wird jedem $t \in T$ durch $D : T \rightarrow \mathcal{D}$ sein Dienstnetz zugewiesen, wobei $(\mathcal{D}, D^{\mathcal{D}}(R))$ eine Dienstklasse sein muss (vgl. Def. 31).

Wir fordern, dass die Rollen des Profils im Nachbereich einer Transition disjunkt sind. Dies erlaubt es, eine das Dienstnetz $D(t)$ eindeutig in seine R-Komponenten zu zerlegen (*Zerlegungseigenschaft*). Relevant sind alle Rollen $R(p)$ mit $p \in t^\bullet$, während die Komponente der restlichen Rollen $R(D(t)) \setminus R(t^\bullet)$ an anderer Stelle implementiert wird. Insgesamt ergibt sich die Zerlegung:

$$\mathcal{E}(t) = \{R(p) \mid p \in t^\bullet\} \cup \{R(D(t)) \setminus R(t^\bullet)\} \tag{3.3}$$

Teamnetze sind R/D-Netze, deren Grundstruktur ein Kausalnetz bildet, das mit einer Stelle initialisiert ist und nur mit Transitionen endet. Abbildung 3.6 zeigt ein Teamnetz.

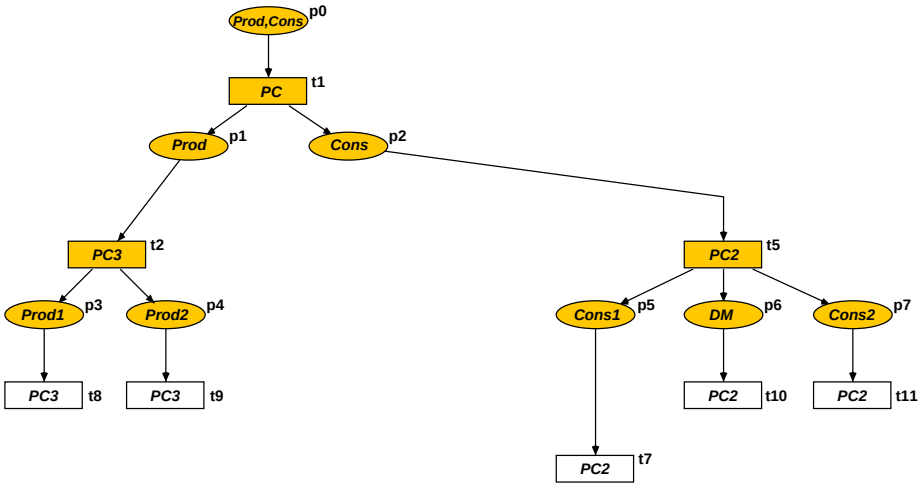


Abbildung 3.6: Ein Teamnetz

Definition 36 *Ein* R/D-Netz $RD = (N, R, D)$ *besteht aus den folgenden Komponenten:*

1. $N = (P, T, F)$ *ist ein Delegationsnetz.*
2. $R : P \to \mathcal{R}$ *ist die Rollenabbildung.*
3. $D : T \to \mathcal{D}$ *ist die Dienstnetzabbildung.*

Ein R/D-Netz heißt Teamnetz*, wenn gilt:*

1. N *ist ein stark zusammenhängendes Kausalnetz.*
2. N *besitzt genau einen minimalen Knoten:* $|{}^\circ N| = 1$.

Für ein R/D-Netz $RD = (N, R, D)$ *definieren wir die Menge der Stellen, die die Rolle* R *anstoßen können, als:*

$$P_{RD}(R) = \{p \in P \mid R(p) = R, {}^\bullet p = \emptyset\} \tag{3.4}$$

Ein Dienst D_0 kann vom R/D-Netz $RD = (N, R, D)$ *geleistet* werden, wenn es eine Stelle $p_0 \in P_{RD}(R(D_0))$ gibt, für die eine Transition $t \in p_0{}^\bullet$ existiert, so dass $D(t) = D_0$ gilt.

Teamnetze notieren wir auch in der Form (K, R, D) mit $K = (B, E, F)$, weil dies für Kausalnetze eine übliche Darstellungsweise ist. Für ein beliebiges R/D-Netz $RD = (N, R, D)$ mit $N = (P, T, F)$ definieren wir die Abbildung $R : 2^P \to \mathcal{R}$ durch $R(P') = \bigcup\{R(p) \mid p \in P'\}$.

Lemma 2 *Elementare Eigenschaften von R/D-Netzen:*

1. *R/D-Netze sind Free-Choice Netze.*
2. *Für alle Teamnetze gilt* $\forall p \in P : |p^\bullet| = 1$ *und* $\forall t \in T : |{}^\bullet t| = 1$.
3. *Alle minimalen Knoten sind Stellen und alle maximalen Knoten sind Transitionen:* ${}^\circ N \subseteq P$ *und* $N^\circ \subseteq T$.

4. *Jedes Teamnetz N besitzt genau einen Platz als minimalen Knoten* $^\circ N \subseteq P$.

5. *Wenn p eine Stelle des Teamnetzes (N, R, D) mit $N = (P, T, F)$ ist, dann ist auch das auf $P_p = (\uparrow p \cap P)$ und $T_p = (\uparrow p \cap T)$ eingeschränkte Netz ein Teamnetz:*

$$RD_{\Uparrow p} := (N_p, R_{|P_p}, D_{|T_p}) \quad mit \quad N_p = (P_p, T_p, F \cap (P_p \cup T_p)^2) \tag{3.5}$$

Für jedes $p \in P$ bezeichnen wir $RD_{\Uparrow p}$ als das Subteam *von RD.*

Beweis: Zu (1): Dies gilt wegen der Bedingung $|{}^\bullet t| = 1$.

Zu (2): Teamnetze sind R/D-Netze, für die nach Definition $|p^\bullet| > 0$ gilt. Da Teamnetze Kausalnetze sind, gilt außerdem $|p^\bullet| \leq 1$. Insgesamt folgt $|p^\bullet| = 1$. Nach Definition gilt für R/D-Netze $|{}^\bullet t| = 1$.

Zu (3): Mit (2) gilt $|p^\bullet| = 1$, also können maximale Knoten keine Stellen sein. Wegen $|{}^\bullet t| = 1$ können analog minimale Knoten keine Transitionen sein.

Zu (4): N besitzt genau einen minimalen Knoten, der nach (3) eine Stelle ist.

Zu (5): N_p ist als Teilnetz wieder ein Kausalnetz mit p als der einzigen initialen Stelle. q.e.d.

Um den besonderen Stellenwert des minimalen Knoten hervorzuheben, wird ein Teamnetz (N, R, D) mit dem minimalen Platz p_0 auch als $R(p_0)$-Team bezeichnet.

3.2.1 Statische Wohlgeformtheit

Betrachten wir nun die Konsistenz eines R/D-Netzes. Diese ist nur dann gegeben, wenn Dienstnetze und Rollenprofile zur Struktur des Teams passen. Sei $(\mathcal{D}, D^{\mathcal{D}}(R))$ eine Dienstklasse. Jede Rolle R besitzt den mit $D^{\mathcal{D}}(R)$ bezeichneten Referenzdienstnetz. Für den Referenzdienst von R gilt offensichtlicht:

$$R \subseteq R(D^{\mathcal{D}}(R))$$

Dies gilt insbesondere für $R = R(p(t))$.

Sei t eine delegierende Tätigkeit, d.h. $t^\bullet \neq \emptyset$. Wir fordern in diesem Fall $|t^\bullet| > 1$, da $|t^\bullet| = 1$ bedeutet, dass eine Aufgabe nur von einer Rolle zur nächsten gereicht wird. Der Delegationsprozess ist nicht produktiv – er stottert, und dies ist nicht erwünscht. Alle Rollen der Profile des Nachbereichs $t^\bullet$ müssen Teil des Dienstes $D(t)$ sein:

$$R(t^\bullet) \subseteq R(D(t))$$

Die Rollen des Nachbereich, d.h. $R(t^\bullet)$ realisieren die Rollen $R(p(t))$. Die restlichen Rollen, d.h. $R(D(t)) \setminus R(t^\bullet)$ und $R(D(R(p(t)), \mathcal{D})) \setminus R(p(t))$ müssen dagegen unverändert sein:

$$R(D(R(p(t)), \mathcal{D})) \setminus R(p(t)) = R(D(t)) \setminus R(t^\bullet)$$

Die Rollen $R(t^\bullet)$ des Dienstes $D(t)$ müssen zudem noch eine Verfeinerung des Rollen $R(p(t)) = R({}^\bullet t)$ Referenzdienstes $D^{\mathcal{D}}(R(p(t)))$ sein:

$$\langle\!\langle D(R(p(t)), \mathcal{D}); R({}^\bullet t), R(t^\bullet); D(t) \rangle\!\rangle$$

Der Dienst $D(t)$ einer Transition t muss also nicht direkt zur Rolle $R(p(t))$ passen. Es reicht, wenn $D(t)$ verhaltensäquivalent zu $R(p)$ ist.

Ist t dagegen eine ausführende Tätigkeit, d.h. $t^\bullet = \emptyset$, dann ist nur gefordert, dass die Rolle des Vorbereichs zum Dienst passt: $R(p(t)) \subseteq R(D(t))$.

Definition 37 *Ein R/D-Netz (N, R, D) ist* wohlgeformt*, wenn die folgende Bedingungen erfüllt sind:*

1. *Rollenpartition: Die Rollenabbildung $R : P \to \mathcal{R}$ erfüllt die Zerlegungseigenschaft:*
$$\forall t \in T : \forall p, p' \in t^\bullet : p \neq p' \Longrightarrow R(p) \cap R(p') = \emptyset$$

2. *Delegation: Für alle Transitionen $t \in T_N$ mit $t^\bullet \neq \emptyset$ muss folgendes gelten:*
$$\begin{array}{rcll} |t^\bullet| & > & 1 & \textit{(Stotterfreiheit)} \\ R(t^\bullet) & \subseteq & R(D(t)) & \textit{(Rollenkompatibilität)} \\ R(D_t) \setminus R(^\bullet t) & = & R(D(t)) \setminus R(t^\bullet) & \textit{(Kontext)} \\ \multicolumn{3}{c}{\langle\!\langle D_t; R(^\bullet t), R(t^\bullet); D(t) \rangle\!\rangle} & \textit{(Verhaltensverfeinerung)} \end{array}$$
Hierbei bezeichnet $D_t := D^{\mathcal{D}}(R(^\bullet t))$ das Referenzdienstnetz zur Rolle $R(^\bullet t)$.

3. *Ausführung: Für alle Transitionen $t \in T_N$ mit $t^\bullet = \emptyset$ muss gelten:*
$$R(p(t)) \subseteq R(D(t)) \qquad \textit{(Rollenkompatibilität)}$$

Beachte, dass jedes Subteam $RD_{\restriction p}$ mit $p \in P_N$ eines wohlgeformten R/D-Netzes (N, R, D) dies auch ist.

In R/D-Netz sind Stellen mit leerem Vorbereich erlaubt. Eine wünschenswerte Eigenschaft solche Stellen ist, dass ihr Rollenprofil $R(p)$ mit den Rollen des Dienstnetz $D(t)$ für alle $t \in p^\bullet$ übereinstimmt:

$$^\bullet p = \emptyset \Longrightarrow \forall t \in p^\bullet : R(p) = R(D(t)) \tag{3.6}$$

Dies ist sinnvoll, da eine Stelle mit leerem Vorbereich den Beginn einer Aufgabenbearbeitung darstellt. Es ist daher nicht sinnvoll, dass ein Dienst mehr Rollen involviert. Weniger Rollen sind ohnehin nicht sinnvoll, da diese dann die Aufgabe von p nicht realisieren können. Wir wollen diese Eigenschaft aber für die Wohlgeformtheit dennoch nicht fordern, da sie nur für die globale Sichtweise auf eine Organisation sinnvoll ist (vgl. dazu auch Abschnitt 4.3). Wir wollen aber auch noch Suborganisationen, d.h. Unterstukturen von Organisationen betrachten, und für diese kann die Eigenschaft im allgemeinen nicht erfüllt sein.

Lemma 3 *Eine Transition $t \in T$ heißt* vollständig*, wenn $R(t^\bullet) = R(D(t))$ gilt, ansonsten* unvollständig*.*

Sei (N, R, D) ein wohlgeformtes R/D-Netz, dann gilt $R(p(t)) = R(D^{\mathcal{D}}(R(p(t))))$ für jede vollständige Transition $t \in T_N$.

Beweis: Für eine vollständige Transition t gilt $D(t) \setminus R(t^\bullet) = \emptyset$ und wegen der Kontextbedingung verschärft sich die Inklusion $R(p(t)) \subseteq R(D^{\mathcal{D}}(R(p(t))))$ zur Gleichheit $R(p(t)) = R(D^{\mathcal{D}}(R(p(t))))$. q.e.d.

Beispiel Abbildung 3.4 zeigt ein Teamnetz. Die Dienstnetze und die Rollen sind im Teamnetz aus Abbildung 3.4 bereits als Anschriften enthalten. In dieser Organisation wird die Rolle „Consumer“ nicht direkt besetzt, denn p_2 ($Cons$) verweist nicht auf eine terminale Transition. Die Aufgaben p_3 ($Prod_1$) und p_4 ($Prod_2$) werden dagegen direkt implementiert.

Wir betrachten eine Organisation mit den Diensten PC, PC_2 und PC_3. Die Dienste sind in Abbildung 2.1, 2.8 und 2.7 dargestellt. Man erkennt, dass PC_2 eine Verfeinerung des Producer/Consumer Dienstes PC bezüglich der Rolle *Cons* und PC_3 eine Verfeinerung bezüglich der Rolle *Prod* ist.

$$\begin{array}{rcl} \mathcal{R} & = & 2^{Rol} \setminus \{\emptyset\} \quad \text{mit} \quad Rol = \{Prod, Cons, DM, Cons_1, Cons_2, Prod_1, Prod_2\} \\ \mathcal{D} & = & \{PC, PC_2, PC_3\} \end{array}$$

Die Referenzdienste sind:

$$D^{\mathcal{D}}(Prod) = D^{\mathcal{D}}(Cons) = PC$$

Die Rollen der Dienste sind:

$$\begin{array}{rcl} R(PC) & = & \{Prod, Cons\} \\ R(PC_2) & = & \{Prod, DM, Cons_1, Cons_2\} \\ R(PC_2) & = & \{Prod_1 Prod_2, Cons\} \end{array}$$

Die Profile realisieren die Rollenprofile $R(P) = \mathcal{R}$ und überdecken daher die Rollenmenge der Dienste $R(PC) \cup R(PC_2)$.

Wir überprüfen die notwendige Implementierungs- und Verfeinerungsbeziehung für die Transition t_1. Es gilt $p(t_1) = p_0$. Die Rollenkompatibilität ist:

$$R(t_1{}^\bullet) = \{Prod, Cons\} \subseteq R(D(t_1)) = \{Prod, Cons\}$$

Da $R(\{p_1, p_2\}) = R(D(t_1))$ gilt, ist t_1 vollständig. Der Kontext bleibt erhalten:

$$R(D(R(p_0), \mathcal{D})) \setminus R(p_0) = \emptyset = R(D(t_1)) \setminus R(t_1{}^\bullet)$$

Für die Verhaltensverfeinerung ergibt sich:

$$\langle\!\langle D(R(p_0), \mathcal{D}); R(p_0), R(t_1{}^\bullet); D(t_1) \rangle\!\rangle = \langle\!\langle PC; \{Prod, Cons\}, \{Prod, Cons\}; PC \rangle\!\rangle$$

Für t_2 folgt aus $p(t_2) = p_1$, $R(p_1) = \{Prod\}$ und $R(t_2{}^\bullet) = \{Prod_1, Prod_2\}$ die erste Rolleninklusion

$$R(t_2{}^\bullet) = \{Prod_1, Prod_2\} \subseteq R(D(t_2)) = \{Prod_1, Prod_2, Cons\}$$

und auch der Kontext:

$$R(D(R(p_1), \mathcal{D})) \setminus R(p_1) = \{Cons\} = R(D(t_2)) \setminus R(t_2{}^\bullet)$$

Da PC_3 eine Verfeinerung der Rolle *Prod* für PC ist, ergibt sich die Verhaltensverfeinerung:

$$\langle\!\langle D(R(p_1), \mathcal{D}); R(p_1)), R(t_2{}^\bullet); D(t_2) \rangle\!\rangle = \langle\!\langle PC_3; \{Prod\}, \{Prod_1, Prod_2\}; PC \rangle\!\rangle$$

Analog für t_3 und t_4. Für t_5 ergibt sich mit $p(t_5) = p_2$ und $R(t_5{}^\bullet) = \{Prod, DM, Cons_1, Cons_2\}$ die Rollenkompatibilität:

$$\begin{array}{rcl} R(D(R(p_2), \mathcal{D})) \setminus R(p_2) & = & \{Prod, Cons\} \setminus \{Cons\} = \{Prod\} \\ & = & \{Prod, DM, Cons_1, Cons_2\} \setminus \{DM, Cons_1, Cons_2\} \\ & = & R(D(t_5)) \setminus R(t_5{}^\bullet) \end{array}$$

Da PC_2 eine Verfeinerung der Rolle *Cons* für PC ist, ergibt sich die Verhaltensverfeinerung zu:

$$\langle\!\langle D(R(p_2, \mathcal{D}); R(p_2), R(t_5{}^\bullet); D(t_5)) \rangle\!\rangle = \langle\!\langle PC; R(p_2), R(t_2{}^\bullet); PC_2 \rangle\!\rangle.$$

Insgesamt zeigt sich, dass das R/D-Netz wohlgeformt ist. ◇

Die für wohldefinierte Teamnetze lokal gefordert Rollenverfeinerung erweitert sich auf die gesamte Struktur. Für einen Stellemenge C eines Teamnetzes (N, R, D) definiere die folgende Komposition:

$$D[C] := D[R(C)] := \Big\|_{p \in C} D({}^\bullet p)[R(p)] \tag{3.7}$$

Lemma 4 *Sei (N, R, D) ein wohlgeformtes Teamnetz mit der initialen Stelle p_0. Dann gilt, dass das Dienstnetz $D[C]$ für jeden Stellenschnitt $C \neq \{p_0\}$ wohldefiniert ist, und die Rollen $R(C)$ eine Verfeinerung von $R(p_0)$ bilden:*

$$\langle\!\langle D_0; R(p_0), R(p_0); D[C] \rangle\!\rangle \qquad \textit{mit } D_0 := D^{\mathcal{D}}(R(p_0))$$

Beweis: Beweis per Induktion über die Tiefe des Teamnetzes.

Induktionsanfang: Der Induktionsanfang bildet das kleinstmögliche Teamnetz, das aus einer Stelle p_0 besteht, die mit der einzigen Transition t mit einer Kante verbunden ist. Der einzige Schnitt ist $C = \{p_0\}$. Also muss nichts gezeigt werden.

Induktionsschritt: Für den Induktionsschritt nutzen wir aus, dass jedes Teamnetz N durch ein Netz N_0 gebildet wird, das aus einer einzigen Transition t mit ${}^\bullet t = \{p_0\}$ und $t^\bullet = \{p_1, \ldots, p_k\}$ besteht, und den Subteams $N_1, \ldots, N_k$, deren initiale Stelle jeweils die $p_1, \ldots, p_k$ sind. Jeder Stellenschnitt $C \neq \{p_0\}$ setzt sich aus Schnitten C_i der Subteams zusammen: $C = \bigcup_{i=1}^k C_i$.

Da die Subteams alle kleiner sind, gilt für sie die Induktionsannahme, d.h. alle $D[C_i]$ sind wohldefiniert und die $R(C_i)$ realisieren in $D[C_i]$ jeweils $R(p_i)$.

Die $R(C_i)$ müssen disjunkt sein, denn wären sie dies nicht, dann würde eine ihnen gemeinsame Rolle R aufgrund der Zerlegungseigenschaft verschiedene Rollen verfeinern, was nicht sein kann. Die Komposition $D[C]$ ist damit wohldefiniert:

$$\begin{aligned} D[C] &= D\left[\bigcup_{i=1}^k C_i\right] = D\left[R\left(\bigcup_{i=1}^k C_i\right)\right] = D\left[\bigcup_{i=1}^k R(C_i)\right] \text{ mit Thm. 12} \\ &= \Big\|_{i=1}^k D[R(C_i)] = \Big\|_{i=1}^k D[C_i] \end{aligned}$$

Der Dienst $D[C]$ realisiert die Rollen:

$$\bigcup_{i=1}^k R(p_i) = R\left(\bigcup_{i=1}^k \{p_i\}\right) = R(t^\bullet)$$

Nach Definition 37 gilt für t die Verhaltensverfeinerung, d.h. $R(t^\bullet)$ realisiert $R({}^\bullet t) = R(p_0)$. Insgesamt gilt damit, dass $D[C]$ die Rolle $R(p_0)$ realisiert. q.e.d.

Diese Eigenschaft gilt offensichtlich auch, wenn (N, R, D) ein Subteam eines größeren R/D-Netzes ist, da jedes Subteam ebenfalls wohlgeformt ist.

Dass $D[C]$ eine Verfeinerung ist, heißt insbesondere, dass die Menge der Kommunikationskanäle durch C disjunkt aufgeteilt wird. Für wohlgeformte Teams ist es ausgeschlossen, dass sich Rollen zyklisch für ihre Arbeit heranziehen.

Theorem 25 *Sei (N, R, D) ein wohlgeformtes Teamnetz mit der initialen Stelle p_0. Sei $p \in P$ eine Stelle mit $R(p) = R(p_0)$, dann gilt $p = p_0$.*

Beweis: Sei $C_0 = \{p_0\}$ und sei C_1 ein Stellenschnitt mit $p \in C$.

Nach Lemma 4 realisieren sowohl $D[C_0]$ als auch $D[C]$ die Rolle $R(p) = R(p_0)$. Sie besitzen somit auch die gleichen externen Kommunikationskanäle.

Daraus folgt, dass $D[R(\{p\})]$ und $D[R(C \setminus \{p\})]$ keinerlei Kommunikationskanäle teilen.

Da nach Definition 10 jedes Dienstnetz D für jedes Rollenzerlegung $R, (R(D) \setminus R)$ mit $R \neq \emptyset$ und $(R(D) \setminus R) \neq \emptyset$ interne Kanäle besitzen muss, ist dies nur möglich, wenn $C = \{p\}$ gilt. Dann muss aber $p = p_0$ gelten, da wohlgeformte Teamnetze nicht stottern. q.e.d.

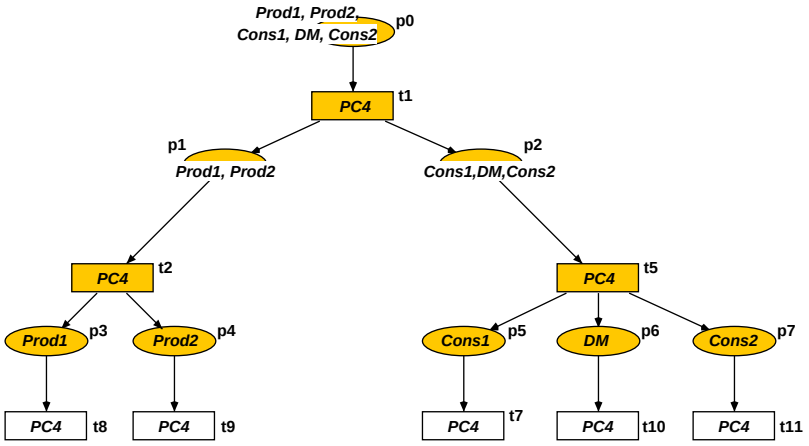


Abbildung 3.7: Ein homogenes Team

Ein besonderer Fall liegt vor, wenn das Teamnetz homogen ist, d.h. wenn alle Dienstnetze $D(t)$ identisch sind. Abbildung 3.7 zeigt ein die homogene Variante des nicht homogenen Teamnetzes aus Abbildung 3.6. In diesem Fall vereinfacht sich Definition 37, denn homogene wohlgeformte Teamnetze haben die Eigenschaft, die zu implementierenden Rollen nur aufzuteilen, nicht aber zu verfeinern.

Lemma 5 *Ein R/D-Netz (N, R, D) heißt* homogen*, wenn $D(t_1) = D(t_2)$ für alle $t_1, t_2 \in T_N$ gilt. Wenn (N, R, D) ein wohlgeformtes Teamnetz ist, dann gilt für alle Transitionen $t \in T$, dass $\{R(p) \mid p \in t^\bullet\}$ eine Mengenpartition von $R(^\bullet t)$ gilt.*

Beweis: Da (N, R, D) ein wohlgeformtes Teamnetz ist, gilt die Verhaltensverfeinerung $\langle\langle D_t; R(^\bullet t), R(t^\bullet); D(t)\rangle\rangle$. Draus folgt insbesondere $R(^\bullet t) = R(t^\bullet)$. Mit der Zerlegungseigenschaft folgt zudem noch, dass $\{R(p) \mid p \in t^\bullet\}$ disjunkte Menge sind. q.e.d.

3.2.2 Teamprozesse

Bezeichne $\mathcal{K}^{pg}(N, m)$ die Menge aller endlichen Prozesse von (N, m) unter der Fortschrittsannahme. Hierbei steht noch nicht fest, ob für ein gegebenes Delegationsnetz ein endlicher Prozess unter der Fortschrittsannahme überhaupt existiert. So könnte es sein, dass unter der Fortschrittsannahme keine endlichen Prozesse generiert werden. So zeigt Abbildung 3.8 ein Netz, das für $m = \{p_1\}$ gar keine endlichen Prozesse (N, m) unter der Fortschrittsannahme besitzt.

Theorem 26 *Sei N ein Delegationsnetz.*

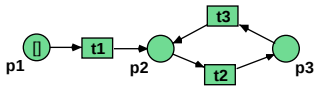


Abbildung 3.8: Ein R/D-Netz ohne Tätigkeitspfade

1. *Ein Prozess* $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ *besteht aus* $|m|$ *disjunkten Teilprozessen.*
2. *Die Prozesse* $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ *von Delegationsnetzen sind Bäume.*
3. *Jeder Prozess korrespondiert zu einem Ableitungsbaum in* $G(n, m)$*.*

Beweis: Dies folgt aus der Linearitätseigenschaft, denn jede Marke wird disjunkt entwickelt.

Prozesse sind als Kausalnetze bereits azyklisch, im allgemeinen aber keine Bäume. Die Baum-Eigenschaft folgt aus der Struktur der Transitionen, genauer aus $|{}^\bullet t| = 1$.

Da die Struktur der Grammatik $G(N, m)$ das Netz direkt abbildet, korrespondieren Prozesse und Ableitungsbäume. q.e.d.

Theorem 27 *Sei* N *ein Delegationsnetz in der Markierung* m *und sei* $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ *ein Prozess, dann gilt:*

1. $\phi(K^\circ) \subseteq T$*.*
2. *Jede Folge* $\phi(w)$ *mit* $({}^\circ K \cap B) \xrightarrow{w} (K^\circ \cap B)$ *ist ein Tätigkeitspfad.*
3. *Mindestens eine Schaltfolge* $\phi(w)$ *mit* $({}^\circ K \cap B) \xrightarrow{w} (K^\circ \cap B)$ *wird von der Grammatik* $G(N, m)$ *erzeugt.*
4. $\mathcal{K}^{pg}(N, \{p\})$ *ist genau dann eine nicht-leere Menge, wenn* A_p *produtiv in* $G(N, m)$ *ist.*

Beweis:

1. $\phi(K^\circ) \subseteq T$ ist äquivalent zu $K^\circ \subseteq E$. Da N keine Senken besitzt ($p^\bullet \neq \emptyset$), gibt es zu jeder Stelle mindestens eine Transition im Nachbereich. Wäre eine Stelle b in K°, dann wäre eine dieser Transition aktiviert – im Widerspruch zu der Annahme, dass K ein Prozess von (N, m) unter der Fortschrittsannahme ist.

2. Da $\phi^\sharp({}^\circ K) = m$ und $\phi(K^\circ) \subseteq T$ gilt haben wir $m = \phi^\sharp({}^\circ K) \xrightarrow{\phi(w)} \phi^\sharp(K^\circ \cap B) = \phi^\sharp(\emptyset) = \mathbf{0}$, d.h. das $\phi(w)$ ein Tätigkeitspfad ist.

3. Jede Ableitung in der Grammatik $G(N, m)$ beschreibt mindestens einen Ablauf in N, wenn auch nicht alle möglichen Permutationen nebenläufiger Ereignisse betrachtet werden.

4. Nach Theorem 24 ist die Variable A_p genau dann produktiv, wenn $\mathbf{0}$ von $m = \{p\}$ erreichbar ist. Dies gilt genau dann, wenn es einen Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ gibt, der $\phi(K^\circ) \subseteq T$ erfüllt, was nach (1) genau dann gilt, wenn überhaupt ein Prozess existiert, d.h. wenn $\mathcal{K}^{pg}(N, m) \neq \emptyset$.

q.e.d.

3.2.3 Bearbeitbarkeit in R/D-Netzen

Die Bedingung, dass eine Stelle sicher bearbeitbar ist, ist verwandt mit der Korrektheitsbedingung für Workflownetze. Im Gegensatz zu korrekten Workflownetzen sind R/D-Netze im allgemeinen jedoch nicht beschränkt. Daher scheidet eine Analyse des Erreichbarkeitsgraphen aus, da dieser im allgemeinen nicht endlich ist. Wir nutzen stattdessen die speziellen Eigenschaft von Delegationsnetzen, um die Analyse der Tätigkeitspfade zu vereinfachen, hier insbesondere die Linearitätseigenschaft von Delegationsnetzen.

Theorem 28 *Sei $RD = (N, R, D)$ ein R/D-Netz und m seine Markierung.*

1. *$TP(m)$ ist genau dann eine nicht-leere Menge, wenn m bearbeitbar ist.*
2. *Ist die Marking m sicher bearbeitbar, dann ist sie auch bearbeitbar.*
3. *N ist genau dann sicher bearbeitbar, wenn es bearbeitbar ist.*
4. *Die Markierung m ist in N bearbeitbar genau dann, wenn alle Markierungen $\{p\}$ mit $m(p) > 0$ bearbeitbar sind.*
5. *Es ist in $O(|N|)$ entscheidbar, ob die Markierung m bearbeitbar ist.*
6. *Es ist in $O(|N|)$ entscheidbar, ob die Markierung m sicher bearbeitbar ist.*
7. *Sind alle Stellen $p \in P$ bearbeitbar, dann ist N sicher bearbeitbar.*
8. *Ist N sicher bearbeitbar, dann ist N nicht (strukturell) lebendig.*

Beweis:

1. Direkt aus der Definition der Bearbeitbarkeit.
2. Folgt aus Definition mit $m \in RS(m)$.
3. Offensichtlich gilt, dass N bearbeitbar ist, wenn es sicher bearbeitbar ist. Ist N bearbeitbar, dann folgt für alle Markierungen, also insbesondere auch für die erreichbaren Markierungen $m' \in RS(m)$ Bearbeitbarkeit. Also folgt, dass N sicher bearbeitbar ist, wenn es bearbeitbar ist.
4. Mit der Linearitätseigenschaft $RS(m_1 + m_2) = RS(m_1) + RS(m_2)$ der Delegationsnetze (Theorem 22) folgt:
$$RS(m) = RS\left(\sum_{p \in P} m(p) \cdot p\right) = \sum_{p \in P} m(p) \cdot RS(\{p\})$$
Also sind die erreichbaren Markierungen durch die Mengen $RS(\{p\})$ charakterisiert.
5. Mit (3) ist die Markierung m bearbeitbar genau dann, wenn alle Markierungen $\{p\}$ mit $m(p) > 0$ bearbeitbar sind. Nach Theorem 24 ist die Nullmarkierung $\mathbf{0}$ von $m = \{p\}$ in N genau dann erreichbar, wenn das Nonterminal A_p der kontextfreien Grammatik $G(N, m)$ produktiv ist. Um Produktivität zu entscheiden, sind die Mengen $PV_n(G)$ zu bilden, wobei n höchstens bis zur Anzahl der Nonterminale wächst. In $G(N, m)$ gibt es zu jeder Stelle p genau ein Nonterminal A_p. Bei der Konstruktion von $PV_n(G)$ sind alle Regeln zu beachten. In $G(N, m)$ gibt es zu jeder Transition t genau eine Regel. Also ist der Aufwand von der Ordnung $|P| \cdot |T| = |N|$.

6. Mit (3) ist die Markierung m sicher bearbeitbar genau dann, wenn alle Markierungen $\{p\}$ mit $m(p) > 0$ sicher bearbeitbar sind. Nach Theorem 24 ist die Markierung $m = \{p\}$ genau dann sicher bearbeitbar, wenn alle erreichbaren Variablen A_p der kontextfreien Grammatik $G(N, m)$ produktiv sind. Die produktiven Variablen $PV_n(G)$ sind nach (4) mit $O(|N|)$ Zeitaufwand zu berechnen. Die erreichbaren Variablen sind durch die Konstruktion der monoton wachsenden Folge $RV_n(G)$ zu bilden, wobei n höchstens bis zur Anzahl der Nonterminale wächst. Analog zu der Menge der produktiven Variablen ist der Konstruktionsaufwand von der Ordnung $|P| \cdot |T| = |N|$. Insgesamt ist der Zeitaufwand also von der Ordnung $O(|N|)$.

7. Sind alle Stellen $p \in P$ bearbeitbar, dann sind alle Markierungen bearbeitbar, insbesondere alle $m' \in RS(m)$.

8. Da stets die Nullmarkierung erreichbar ist, kann das Netz stets in einen Deadlock geraten, also ist es nicht lebendig. Da dies für jede Anfangsmarkierung gilt, ist die Eigenschaft sogar strukturell.

Dies zeigt die Eigenschaften. q.e.d.

Zirkuläre Ausführungspfade existieren genau dann, wenn die Grammatik $G(N, m)$ Iterationen zulässt.

Theorem 29 *Sei (N, R, D) ein R/D-Netz und m seine Markierung. Es ist entscheidbar, ob N keine, endlich oder unendlich viele Tätigkeitspfade besitzt.*

Beweis: Dies gilt genau dann, wenn $L(G(N, m))$ leer, endlich oder unendlich ist. Diese Fragen sind für kontextfreie Grammatiken entscheidbar. q.e.d.

3.3 Koordination von Teams

Bei der Ausführung von Diensten werden Teilaufgaben entlang des Tätigkeitspfades weitergereicht, bis sie bei den terminalen Transitionen ankommen. Im Prozess in Abbildung 3.6 sind das t_8, t_{10}, t_{11}, t_{12} und t_{13}. Durch die Netzstruktur ist sichergestellt, dass die Agenten, die diesen Transitionen zugeordnet sind, die Interaktion gemäß des Dienstes PC_2 ausführen können. Der gesamte Prozess dient aber nur dazu, eine einzige Aktivität zu realisieren, nämlich die durch t_1 angestoßen wurde, d.h. $D(t_1)$. Man könnte meinen, dass die vor den terminalen Transitionen liegenden Ebenen (im Beispiel sind dies t_1, t_2, t_5 und t_7) keinerlei Funktion haben, außer dass sie entscheiden müssen, an wen sie die Aufgaben delegieren. Dies ist nicht ganz so, denn die Aufgabe dieser Agenten besteht unter anderem darin, globale Gültigkeitsbedingungen (wie z.B. Termination der Dienstnetze o.ä.) entlang der Tätigkeitspfade zu garantieren.

3.3.1 Steuerung

Jeder Transition wird durch die Abbildung $\psi : T \to PF_{\mathcal{D}}$ eine Prozessformel zugewiesen, die die Menge der *zugelassenen Prozessen* definiert. Wir unterscheiden hier zugelassene Prozesse von korrekten. Korrektheit spricht hier die formale Spezifikation an, während zugelassene Prozesse mit einer Form der Steuerung zu realisieren sind. Die Steuerung ψ ist der technischen Ausdruck der organisationalen

Sozialstrukturen. In SONAR wird die Steuerung ψ eines Dienstes D durch ein Koordinierungsprotokoll implementiert, das die Schaltmöglichkeiten der Dienstnetze reguliert.

Wir fordern, dass die Abbildung $\psi(t)$ mindestens einen Prozess des Dienstnetzes $D(t)$ erlaubt, d.h. die Steuerung ist realisierbar.

Definition 38 *Ein* koordinierendes Team *ist das Tupel* (N, R, D, ψ)*, wobei gilt:*

- (N, R, D) *ist ein R/D-Netz.*
- $\psi : T_G \to PF_{\mathcal{D}}$ *ist die lokale Steuerung.*

Ein koordinierendes Team (N, R, D, ψ) *ist* wohlgeformt*, wenn* (N, R, D) *dies ist und die Steuerung realisierbar ist:*

$$\forall t \in T : Proc(D(t), \psi(t)) \neq \emptyset$$

Aus der lokalen Steuerung $\psi : T \to PF_{\mathcal{D}}$ ergibt sich für jede Transition $t \in T$ eines koordinierenden Teams die *globale Steuerung* $\hat{\psi}$, indem wir alle F-Vorgänger miteinbeziehen:

$$\hat{\psi}(t) := \bigwedge_{(t',t)\in F^*} \psi(t') \tag{3.8}$$

Man beachte, dass die globale Steuerung $\hat{\psi}(t)$ eine endliche Konjunktion ist, da es nur endliche viele Elemente t' mit $(t', t) \in F^*$ gibt, nämlich höchstens alle $t \in T$.

Wir definieren Koordination hier allgemein für Delegationsstrukturen, werden aber im folgenden nur Kausalnetze betrachten, da wir an Teamprozessen interessiert sind. In einem Kausalnetz beschreibt $\hat{\psi}(t)$ also genau die Konjunktion aller lokalen Steuerungbedingungen von der Initialisierung des Teams bei p_0 bis hin zu t.

Die Steuerung ψ schränkt die Abläufe insofern ein, als dass wir für die Planung eines Agenten nur solche Dienstznetzprozesse des Dienstes $D(t)$ zulassen wollen, die $\hat{\psi}(t)$ erfüllen. Die lokale Steuerung ψ hat die Aufgabe, den akteursbezogenen Planungsprozess vorab einzuschränken, während die globale Steuerung die Handlungswahl eines im Team planenden Agenten noch weiter einschränkt.

3.3.2 Teamdienste

Jedes Teamnetz definiert durch seine maximalen Transitionen K° eine Interaktion. Jede Transition $t \in K^\circ$ führt den Anteil des Dienstes $D(t)$ aus, der durch die Rolle $R(p(t))$ der Stelle im Vorbereich von t definiert ist:

$$D(G) := \Big\|_{t\in K^\circ} D(t)[R(p(t))] \tag{3.9}$$

Aus Lemma 4 folgt, dass $D(G)$ wohldefiniert ist, denn es gilt:

$$\|_{t\in K^\circ} D(t)[R(p(t))]D(G) := \|_{p\in {}^\bullet(K^\circ)} D({}^\bullet p)[R(p)]$$

Für homogene Dienstnetze sind alle Dienstnetze $D({}^\bullet p)$ identisch, so dass $D(G) = D(t)$ für jedes beliebige $t \in T$ gilt.

Für das Teamnetz in Abbildung 3.6 ergibt sich somit der Teamdienst:

$$\begin{aligned} & D(t_8)[R(p_3)]\|D(t_9)[R(p_4)]\|D(t_{13})[R(p_9)]\|D(t_{10})[R(p_8)]\|D(t_{11})[R(p_7)] \\ = \; & PC_3[Prod_1]\|PC_3[Prod_2]\|PC_2[Cons_1]\|PC_2[DM]\|PC_2[Cons_2] \end{aligned}$$

Man beachte, dass diese Interaktion weder PC_2 noch durch PC_3 beschrieben wird, sondern eine Kombination aus beiden darstellt.

Ebenso ergibt sich die Teamsteuerung $\hat{\psi}(G)$ als:

$$\hat{\psi}(G) := \bigwedge_{t \in K^\circ} \hat{\psi}(t) \tag{3.10}$$

Definition 39 *Sei G ein koordiniertes Team, dann ist der* Teamdienst *definiert als $D(G) := \|_{t \in K^\circ} D(t)[R(p(t))]$ und die* Teamsteuerung *definiert durch $\hat{\psi}(G) := \bigwedge_{t \in K^\circ} \hat{\psi}(t)$.*

Rollenkomponenten und Teamdienste sind miteinander verträglich.

Lemma 6 *Sei G ein Team, dann gilt für alle $t \in K^\circ$ und für beliebige Formeln $\phi \in PF_{\mathcal{D}}$:*

$$Proc(D(G), \phi) \subseteq Proc(D(G)) \subseteq Proc(D[R(p(t))])$$

Beweis: Mit Theorem 11 folgt:

$$\begin{aligned} Proc(D(G)) &= Proc(\|_{t \in K_G^\circ} D[R(p(t))]) \\ &= \bigcap_{t \in K_G^\circ} Proc(D[R(p(t))]) \subseteq Proc(D[R(p(t))]) \end{aligned}$$

Die Inklusion $Proc(D(G), \phi) \subseteq Proc(D(G))$ ist offensichtlich. q.e.d.

3.4 Sonar-Agenten und Teamwork

Wir betrachten jetzt die Ausführungsumgebung und deren Handlungslogik, mit deren Hilfe die Organisationale Abläufe und Strukturen implementiert werden: den SONAR-Agenten. Die Team-Koordinierung wird durch die oberste, die sozial koordinierenden Schicht der Agentenarchitektur von Abb. 3.9 vorgenommen.

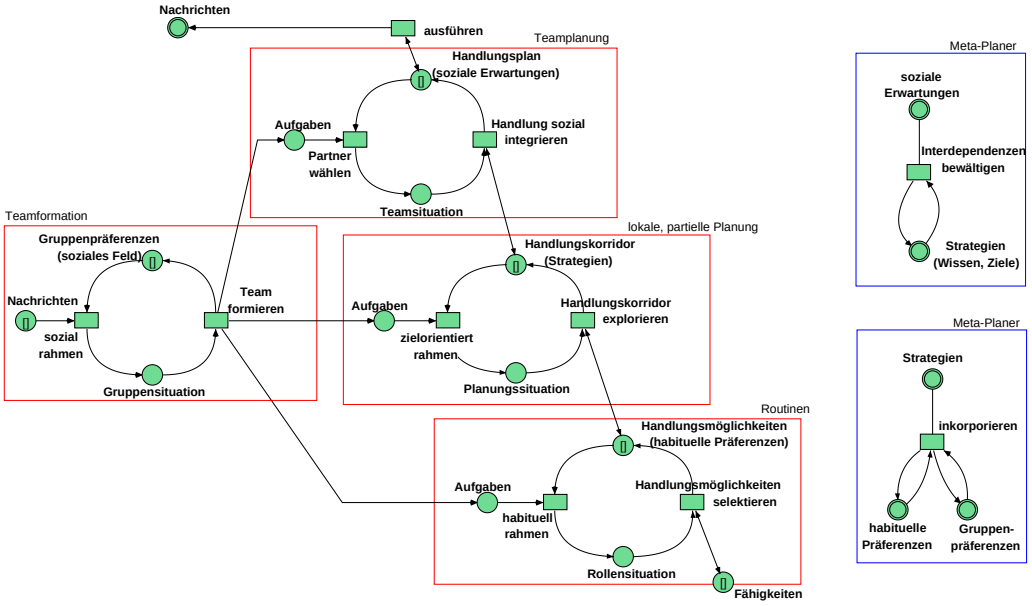


Abbildung 3.9: Die Planhierarchie im SONAR-Agenten

Der SONAR-Agent aus Abbildung 3.9 besitzt drei Ebenen der Planung und zwei darauf operierende Metastrukturen. *Planer* dienen der strategiegesteuerten Exploration des Möglichkeitsraumes eines Dienstes. Berücksichtigt werden zum einen elementare Handlungsstrukturen und die Kompositionsoperatoren hierauf und zum anderen die Explorationssstrategien. Planer regeln das Verhalten der Transitionen: Handlungen sozial integrieren, Handlungskorridor explorieren und Handlungsmöglichkeiten selektieren. Planer bestehen initial aus dem aktuellen Handlungsraum, der sich aus der Historie der eingehenden Nachrichten und der internen Verarbeitung bestimmt. Im Beispiel sind dies die Stellen Handlungsplan, Handlungskorridor und Handlungsmöglichkeiten im SONAR-Agenten. Planer reagieren auf äußere Signale, kontextualisieren diese in bezug auf den Handlungsraum und adjustieren den Handlungsraum ggf. neu. Die Prozesse der Kontextualisierung und der Adjustierung sind abhängig von Parametern, nämlich den Fähigkeiten bezüglich der atomaren Handlungsmöglichkeiten (den Aufspannenden des Handlungsraumes) einerseits und den Adjustierungsstrukturen (habituelle Präferenzen, Strategien und soziale Erwartungsstrukturen) andererseits.

Die *Meta-Planer* dienen der Adaption und Reorganisation der Planer, indem sie die Planungsressourcen modifizieren oder ergänzen, beispielsweise indem Strategien angepasst, Handlungsroutinen erlernt, Agenten gegen andere ausgetauscht oder Koordinierungsregeln generiert werden. Es handelt sich um die beiden Transitionen: Interdependenzen bewältigen und inkorporieren.

Sei $(\mathcal{D}, D^{\mathcal{D}})$ eine Dienstklasse und sei $\mathcal{A}$ eine Menge an Agentennamen. Jeder Agent hat eine Menge von Verhaltensregelmäßigkeiten (VR), mit denen er auf jede Rolle R reagieren kann. Eine *Verhaltensregelmäßigkeit* δ ist ein Paar

$$(D, \{(R_1, A_1), \ldots, (R_n, A_n)\}) \in \mathcal{V}_R(\mathcal{D})$$

mit der Bedeutung, dass die Rolle R vom Dienst D implementiert werden soll, wobei die Rolle R_i nicht selbst übernommen, sondern jeweils an den Agenten A_i delegiert wird. Hierbei verfeinern die Rollen $R_1, \ldots, R_n$ im Dienst D die Rolle R im Dienst D_R. Es ist zu beachten, dass die Rolle R durch jedes $D_R \in \mathcal{D}(R)$ identisch charakterisiert wird, denn alle $D \in \mathcal{D}(R)$ definieren die Rolle auf die gleiche Art und Weise. Die Menge aller Reaktionsschemata zu einer Rolle R ist:

$$\begin{aligned} \mathcal{V}_R(\mathcal{D}) := \{(D, \quad & \{(R_1, A_1), \ldots, (R_n, A_n)\} \mid D \in \mathcal{D}, n \in \mathbb{N}, \\ & \forall D_R \in \mathcal{D}(R) : \langle\langle D; \{R_1, \ldots, R_n\}, R; D_R \rangle\rangle \\ & \textstyle\bigcup_{i=1}^{n} R_i \subseteq R(D), \\ & \forall 1 \leq i, j \leq n : i \neq j \Longrightarrow R_i \cap R_j = \emptyset, \\ & \forall 1 \leq i \leq n : A_i \in \mathcal{A}\} \end{aligned} \tag{3.11}$$

Die Menge aller Verhaltensregelmäßigkeiten ist:

$$\mathcal{V}(\mathcal{D}) := \bigcup_{R \in \mathcal{R}} \mathcal{V}_R(\mathcal{D})$$

Für jedes $\delta = (D, \{(R_1, A_1), \ldots, (R_n, A_n)\}) \in \mathcal{V}(\mathcal{D})$ definiere die Projektionen:

$$D_\Delta(\delta) := D \qquad R_\Delta(\delta) := R_1 \cup \cdots \cup R_n \qquad \mathcal{A}_\Delta(\delta) := \{A_1, \ldots, A_n\} \tag{3.12}$$

Jede Stelle $p \in P$ eines Teams beschreibt eine Aufgabe im Team. Die Transitionen $t \in T$ bearbeiten diese Aufgaben. In einem Gruppennetz wird jeder Tätigkeit t der Agent $A(t)$ zugewiesen, der die Tätigkeit ausführt. Es ist praktisch, auch dem Aufgaben p Agenten zuzuweisen. Wir fordern dabei $A(p) = A(t)$ für $p \in t^\bullet$, d.h. die vom Agenten $A(t)$ generierten Teilaufträge sind ihm auch zugeordnet.

Definition 40 *Ein* Gruppennetz $G = (N, R, D, \psi, A)$ *besteht aus einem Teamnetz* (N, R, D, ψ) *und der Besetzung* $A : (P \cup T) \to \mathcal{A}$ *mit* $A(p) = A(t)$ *für alle* $p \in t^\bullet, t \in T$.

Für eine Menge an Gruppennetze $\mathcal{G}$ *ist die Menge aller Profile definiert als* $\mathcal{P}_\mathcal{G} := \bigcup_{G \in \mathcal{G}} P_G$ *und die Menge aller Tätigkeiten als* $\mathcal{T}_\mathcal{G} := \bigcup_{G \in \mathcal{G}} T_G$.

Ist der Kontext eindeutig, so bezeichnen wir sowohl R/D-, Team- als auch Gruppennetze jeweils als *Team*. Ein Gruppennetz G wird auch als $A(p_0)$-*Team* bezeichnet, wegen der besonderen Bedeutung der initialen Stelle, die durch folgenden Satz charakterisiert wird.

Theorem 30 *Jede Abbildung* $A' : T \to \mathcal{A}$ *eines Teamnetzes wird durch durch die Definition von* $A(p_0)$ *für den minimalen Knoten* p_0 *eindeutig zu einer Besetzung* $A : (P \cup T) \to \mathcal{A}$ *erweitert.*

Beweis: $A(p_0)$ ist definiert. Wir setzen $A(t) = A'(t)$ für alle $t \in T$ und $A(p) = A'(t)$ für alle $p \in t^\bullet$. Somit ist die Eigenschaft der Definition aus Def. 40 erfüllt. Aus der Tatsache, dass N ein Kausalnetz mit $N^\circ \subseteq T$ ist, folgt, dass A für alle Knoten aus $P \cup T$ definiert ist. q.e.d.

3.4.1 Sonar-Agenten

Ein SONAR-Agent besitzt Strukturen zur Teambildung, zur Planung und zur Reorganisation durch Meta-Planung. Die Planung nutzt dabei die Dienstklasse $(\mathcal{D}, D^\mathcal{D}(R))$. Außerdem ist definiert, welche Rollen der Agent einnehmen kann. Die Menge $\rho(D)$ bezeichnet die Menge der Rollen, die der Agent in der Lage ist einzunehmen. Gilt $\rho(D) = \emptyset$, so kann der Agent keine Rolle des Dienstes implementieren. Die Fähigkeiten $\rho(D)$ implizieren zudem, dass der Agent alle in der der Dienstkomponenten $D[\rho(D)]$ benötigten Rechte oder Ressourcen usw. besitzt.

Definition 41 *Sei* $\mathcal{A}$ *eine Menge an Agentennamen,* $(\mathcal{D}, D^\mathcal{D})$ *eine Dienstklasse zur Rollenstruktur* $\mathcal{R}$ *und* $\mathcal{G}$ *eine Menge an Gruppennetzen. Ein* SONAR-Agent

$$(\rho, \gamma, \Delta, \delta, \pi, \beta^1, \beta^2)$$

besteht aus folgenden Komponenten:

1. $\rho : \mathcal{D} \to \mathcal{R}$ *sind die Fähigkeiten, die* $\rho(D) \subseteq R(D)$ *für alle* $D \in \mathcal{D}$ *erfüllen.*
2. $\gamma : \mathcal{R} \to \mathcal{G}$ *ist die Teamgenerationsfunktion.*
3. $\Delta : \mathcal{G} \to \mathcal{P} \to 2^{\mathcal{V}(\mathcal{D})}$ *ist eine Selektionsfunktion der Verhaltensregelmäßigkeit.*
4. $\delta : \mathcal{G} \to 2^{\mathcal{V}(\mathcal{D})} \to \mathcal{P}_\mathcal{G} \to \mathcal{V}(\mathcal{D})$ *ist eine Selektionsfunktion des Handlungskorridors.*
5. $\pi : \mathcal{G} \to \mathcal{D} \to \mathcal{P}_\mathcal{G} \to Proc(\mathcal{D}, \mathcal{R})$ *ist eine Selektionsfunktion der sozialen Integration.*
6. β^1 *ist der Meta-Planer der Inkorporierung.*
7. β^2 *ist der Meta-Planer der Interdependenzbewältigung.*

Die Formationsfunktion γ und die Selektionsfunktionen Δ, δ und π besitzen die im nachfolgenden definierten Eigenschaften (3.13) bis (3.16).

Eine Konfiguration *eines Agenten wird durch die Fähigkeiten ρ, den Teamplaner γ und durch die Selektionsfunktionen Δ, δ und π definiert:*

$$(\rho, \gamma, \Delta, \delta, \pi)$$

Die Menge aller SONAR*-Agenten wird mit $\mathcal{S}_{(\mathcal{A},\mathcal{R},\mathcal{D},\mathcal{G})}$ bezeichnet (auch kurz als $\mathcal{S}$, wenn der Kontext klar ist.).*

Wir erweitern das Modell, indem wir noch einen internen Zustand $z \in Z$ eines Agenten definieren. Jede Auswahl ist dann zustandsabhängig, d.h. statt Δ verwenden wir $\Delta(z)$ usw. Dieser Zustand z bildet die Initialmarkierung des SONAR-Agenten aus Abb. 3.9. Es handelt sich um die Stellen Fähigkeiten, Handlungsmöglichkeiten, Handlungskorridor und Handlungsplan. Jede Auswahl, d.h. jedes Schalten einer Transition in Abb. 3.9 modifiziert den Zustand. Dies kommt durch die Kreise in den Planungsschichten aus Abb. 3.9 zum Ausdruck. Wir werden diese Erweiterung jedoch nicht explizit notieren, sondern gehen im folgenden davon davon aus, dass jede Planungstransition den Zustand modifiziert.

Verteiltes Planen im Team ist in zwei Aspekte unterteilt. In der ersten Phase wird ein Teamplan generiert, in der zweiten Phase wird dieser gemeinsame Plan ausgeführt.

In der formalen Darstellung beschreiben wir Teamwork als das Zusammenspiel von Teamgenerierung und Planung. Die beiden Abläufe sind nicht losgelöst voneinander zu betrachten, das sie durch die Bedingungen an die Wohlkoordiniertheit eng verpflochten sind. Die hier gewählte Darstellung besitzt den Vorzug, dass sie eine klare Spezifikation darstellt. Man darf sie jedoch nicht als Phasen eines Algorithmus misdeuten. Dieser würde zunächst ein Team generieren, dann würden alle Agenten des Teams lokal ihre Handlungen planen und schließlich würde dieser Plan, sofern er wohlkoordiniert ist, ausgeführt. Würde man die so umzusetzen, so erhielte man sicherlich keine effiziente Zusammenarbeit von Agenten. Es sind also die Präferenzen der Beteiligten – schon aus Effizienzgründen – bereits bei der Teamformation zu berücksichtigen.

3.4.2 Formation eines Teams

Sei ein Multiagentensystem gegeben. Die Teamaktivität geht vom Agenten $A_0 \in \mathcal{A}$ aus, der die Teamaktivität anstößt. Das koordinierte Team, das A_0 generiert, muss ein A_0-Team sein. Es hat die Aufgabe die Rolle R_0 auszufüllen. Das Team wird in Abhängigkeit der Teamgenerationsfunktion $\gamma : \mathcal{R} \to \mathcal{G}$ des Agenten gebildet:

$$G_0 = \gamma(R_0) = (K_G, R_G, D_G, \psi_G, A_G) \quad \text{mit} \quad K_G = (P_G, T_G, F_G)$$

Das Team $\gamma(R_0)$ muß für alle $R_0 \in \mathcal{R}$ ein R_0-Team sein, d.h. die Rolle $R_G(p_0)$ der initialen Stelle p_0 des Teams (d.h. ${}^\circ K_G = \{p_0\}$) muss R_0 sein:

$$\forall R \in \mathcal{R} : R_G({}^\circ K_{\gamma(R)}) = R \tag{3.13}$$

Zur Aufgabe p_0 wählt A_0 einen Agenten A_{p_0}, der damit beauftragt wird, diese zu bearbeiten. Dazu wählt A_0 eine Aktivität t_0 sowie den Dienst $D_G(t_0)$ und setzt $A_G(t_0) = A_{p_0}$. Dabei muss D_0 in der Lage sein, die Rolle $R_G(p_0) = R_0$ zu implementieren (vgl. Def. 37). Die Aktivität wird realisiert, indem die Teilaufgaben

$t_0{}^\bullet = \{p_1, \ldots, p_n\}$ generiert werden. Für diese legt $R_G(p_i)$ die jeweils zu implementierende Rolle fest, die dann vom Agenten A_{p_i} zu implementieren ist. Für diese Teilaufgaben wiederholt sich die Auswahl, solange bis keine weiteren Teilaufgaben mehr generiert werden (vgl. dazu auch Abschnitt 4.5.1). A_0 erzeugt also nicht nur seine lokale Aufgabenaufteilung, sondern entwirft bereits einen Plan für die gesamte Teamstruktur, in der auch die Aufgaben für andere Agenten festgelegt sind.

Das Team wird aber genauso von jedem anderen Teammitglied mitgestaltet. Eine Kante $(p,t) \in F$ bedeutet, dass der Agent $A_G(p)$ eine Aufgabe p dem Agenten $A_G(t)$ übertragt. Die Teamformation vollzieht sich bezüglich p für $A_G(t)$ auf die gleiche Art und Weise, wie für A_0 bezüglich p_0. Der Agent $A_G(t)$ generiert also mit seiner Auswahlfunktion $\gamma_{A_G(t)}$ das Subteam:

$$\gamma_{A_G(t)}(R(p))$$

Offensichtlich ist es notwendig, dass das Team zu den Absichten aller beteiligten Agenten des Teams passt, d.h. alle generierten (Sub-)Teams müssen zueinander passen und wohlkoordiniert sein. Bevor wir die Bedingungen in Definition 43 formalisieren, beschreiben wir zunächst den Planungsprozess im Team.

3.4.3 Plangenerierung im Team

Nach der Formation des Teams $G = (K_G, R_G, D_G, \psi_G, A_G)$ mit $K_G = (P, T, F)$ beginnen die Agenten mit der Handlungsplanung. Im Agenten ist der Rolle kein fester Dienst zugewiesen, d.h. es ist für ein Team G nicht statisch definiert, wie der Agent A_p die Rolle $R_G(p)$ realisiert. Dies ergibt sich anhand eines Planungsprozesses. Die Agenten des Teams werden von der Transition team formieren aktiviert, indem jeder Agent $A \in \mathcal{A}^*$, der einer finalen Transitionen des Teams zugeordnet ist, die Menge der ihm im Team durch A_G zugewiesenen Aufgaben berechnet. Diese Agenten realisieren den Teamdienst $D(G)$. Es handelt sich um die Stellen:

$${}^\bullet(K^\circ \cap A_G^{-1}(A))$$

Man beachte, dass aus $K^\circ \subseteq T$ direkt ${}^\bullet(K^\circ \cap A_G^{-1}(A)) \subseteq P$ folgt.

Die Kanten $(p,t) \in F_G$ des Teamnetzes K_G stellen Implementationsbeziehungen dar. Da K_G ein Teamnetz ist, existiert zu jeder Tätigkeit t eine eindeutig bestimmte Stelle p im Vorbereich und zu jeder Stelle p eine eindeutig bestimmte Tätigkeit t im Nachbereich (vgl. Lemma 2). Die Aufgabe p wird von dem Agenten $A_G(t)$ implementiert. Dieser Agent $A_G(t)$ führt dann folgende Berechnung durch (vgl. dazu die Schaltfolgen des Netzes aus Abb. 3.9): Die Transition Handlungsmöglichkeiten selektieren trifft zwischen den Dienstnetzen $\mathcal{D}$ eine Auswahl, um die Rolle $R_G(p)$ auszufüllen. Die habituelle Präferenzordnung Δ generiert dazu eine Menge passender Verhaltensregelmäßigkeiten. Das Ergebnis

$$\Delta := \Delta(G)(p) \subseteq \mathcal{V}(\mathcal{D})$$

markiert dann die Stelle Handlungsmöglichkeiten, die in Abhängigkeit von den Fähigkeiten generiert werden. Da die Wahl in Abhängigkeit von p und nicht von $R_G(p)$ getroffen wird, kann die Auswahl flexibel von der Teamstruktur abhängen. Der Handlungskorridor ergibt sich, indem die Transition Handlungskorridor explorieren mit Hilfe der Strategie δ die Verhaltensregelmäßigkeit

$$\delta := \delta(G)(\Delta)(p)$$

selektiert und damit die Stelle Handlungskorridor markiert. Um Handlungen sozial zu integrieren, muss ein Plan, d.h. ein Prozess π der Komponenten

$$D[R_G(p)] \quad \text{mit} \quad D := D_\Delta(\delta)$$

ausgewählt werden, der zum einen die Teamsteuerung $\hat{\psi}_G(p^\bullet)$ respektiert und zum anderen zu den Plänen aller anderen Teammitglieder passt. Letzteres bedeutet, dass im Team ein gemeinsamer Teamplan π_G ausgehandelt wird (vgl. Definition 44). Die Auswahl leistet die Funktion π, die die sozialen Erwartungsstrukturen zum Ausdruck bringt. Der ausgewählte Prozess

$$\pi(G)(D)(p) \in Proc(D[R_G(p)], \hat{\psi}_G(p^\bullet))$$

ist dann der Handlungsplan der die Teamsteuerung erfüllt.

Handlungsplanung involviert drei verschiededene Klassen von *Sektionsfunktionen*:

1. Eine Selektionsfunktion der Verhaltensregelmäßigkeit ist eine Abbildung $\Delta : \mathcal{G} \to \mathcal{P} \to 2^{\mathcal{V}(\mathcal{D})}$ mit

$$\forall G \in \mathcal{G} : \forall p \in P_G : \Delta(G)(p) \subseteq \mathcal{V}_{R_G(p)}(\mathcal{D}) \tag{3.14}$$

2. Eine Selektionsfunktion des Handlungskorridors ist eine Abbildung $\delta : \mathcal{G} \to 2^{\mathcal{V}(\mathcal{D})} \to \mathcal{P}_\mathcal{G} \to \mathcal{V}(\mathcal{D})$ mit

$$\forall G \in \mathcal{G} : \forall \Delta \in 2^{\mathcal{V}(\mathcal{D})} : \forall p \in P_G : \delta(G)(\Delta)(p) \in \Delta \tag{3.15}$$

3. Eine Selektionsfunktion der sozialen Integration ist eine Abbildung $\pi : \mathcal{G} \to \mathcal{D} \to \mathcal{P}_\mathcal{G} \to Proc(\mathcal{D}, \mathcal{R})$ mit

$$\forall G \in \mathcal{G} : \forall D \in \mathcal{D} : \forall p \in P_G : \pi(G)(D)(p) \in Proc(D[R_G(p)], \hat{\psi}_G(p^\bullet)) \tag{3.16}$$

Die Formationsfunktion γ und die Selektionsfunktionen Δ, δ und π eines SONAR-Agenten besitzen nach Definition stets die obigen Eigenschaften (3.13) bis (3.16).

Anmerkung: In der obigen Modellierung liefern die Formationsfunktion γ und die Selektionsfunktionen Δ, δ und π genau ein Ergebnis, nämlich die beste Wahl. Für eine praktische Implementierung ist meist aber auch die zweitbeste Wahl von Interesse, beispielsweise falls innerhalb einer Gruppe ein Kompromiß gefunden werden muss. Um dies zu modellieren, gibt man statt einer Funktion eine komplette Präferenzordnung an, d.h. $\gamma(R_0)$ wäre eine Ordnung der Menge aller Teams $\mathcal{G}$, $\Delta(G)(p)$ würde in dieser Modellierung eine Ordnung der Menge der Teilmengen von Verhaltensregelmäßigkeiten $2^{\mathcal{V}(\mathcal{D})}$ sein, $\delta(G)(\Delta)(p)$ würde eine Ordnung auf der Menge der Verhaltensregelmäßigkeiten Δ darstellen und $\pi(G)(D)(p)$ eine auf der Prozessmenge $Proc(D[R_G(p)], \hat{\psi}_G(p^\bullet))$. Die Ordnungen kann man als surjektive Abbildung der natürlichen Zahlen in die jeweilige Grundmenge modellieren. Die Selektionsfunktionen ergeben sich dann jeweils als das Minimum bezüglich der jeweiligen Präferenzordnung.

3.4.4 Rekonfiguration der Agenten

Die Meta-Planer der Agenten reorganisieren den Planungsprozess, indem sie die Konfiguration des Agenten modifizieren. Dies ist beispielsweise dann nötig, wenn der Teamplanungsprozess keine Übereinkunft liefert. Wir haben zwei Formen der Meta-Planung. Zum einen ist im Modell aus Abb. 3.9 die Transition inkorporieren vorgesehen. Sie modifiziert mit Hilfe des Meta-Planers β^1 die Rollenfähigkeiten ρ, die Verhaltensregelmäßigkeiten Δ und die Teamformation γ:

$$\beta^1 : \rho \mapsto \rho', \gamma \mapsto \gamma', \Delta \mapsto \Delta'$$

Hierbei besitzt die Modifikation von ρ gegenüber denen von Δ oder γ eine besonderen Stellenwert, denn während eine Veränderung von Δ oder γ bedeutet, dass sich die Präferenzen des Agenten ändern, bedeutet eine Veränderung von ρ, dass der Agent sogar seine Fähigkeiten verändert. Der Agent kann daher auch lernen, bis dato unbekannte Rollen einzunehmen.

Zweitens haben wir noch die Transition Interdependenzen bewältigen, die mit Hilfe des Meta-Planers β^2 die Strategien δ und die Planung π modifiziert:

$$\beta^2 : \delta \mapsto \delta', \pi \mapsto \pi'$$

Die Unterscheidung der beiden entspricht ihrer Position in der internen Verabeitung. Für eine Inkorporation verändern sich die Fähigkeiten des Agenten oder die grundlegenden Verhaltensregelmäßigkeiten. Diese Parameter wirken sehr früh und sehr grundsätzlich auf den gesamten Interaktionsprozess ein. Für eine Interdependenzbewältigung werden dagegen die nachgelagerten Phasen des Planungsprozesses modifiziert. Wir wir in den beiden folgenden Kapiteln feststellen werden korrespondieren die beiden Ebenen dabei auch noch zur Definition der Organisation. Die Veränderung der Organisation eines Multiagentensystems, wie wir sie in Kapitel 5 betrachten, ist eine tiefgreifende Umgestaltung des Systems, und sie wirkt gerade auf die Parameter, die von der Meta-Planung der Inkorporation verändert werden.

Die Meta-Planung muss wieder koordiniert stattfinden. Meta-Planung ist eine Teamaktivitäten und ihre Dienst sind die Organisationstransformationen. Wir betrachten diese Form der Koordinierung von reflexiven Meta-Prozessen in Kapitel 5. Dort geben wir Kriterien an, welche Formen der Meta-Planung verträglich mit den MAS-Strukturen sind.

Die Selbstmodifikation ist ein algorithmisch aufwendiges Verfahren. Es ist daher im allgemeinen notwendig, die Strukturen des Multiagentensysteme so zu gestalten, dass möglichst alle Aufgabe realisierbar und das die Agenten Teams generieren, die vom Multiagentensystem abzeptierbar sind. Im folgenden Kapitel betrachten wir daher Organisation, die Multiagentensystem genau diese Strukturen bereitstellen.

3.5 Sonar-Multiagentensysteme und wohlgeformte Teamplanung

Betrachten wir nun, wie sich SONAR-Agenten zu Multiagentensystemen aggregieren lassen und wie sich innerhalb dieser Systeme wohlgeformte Teamplanung vollzieht.

3.5.1 Sonar-Multiagentensysteme

Im folgenden konstruieren wir ein Multiagentensystem, indem wir jedem Agentennamen A einen Agenten $\mu(A)$ zuordnen. Dieses Multiagentensystem nennen wir

SONAR-*Multiagentensystem*, kurz: SONAR-MAS.

Definition 42 *Sei $\mathcal{A}$ die Menge aller Agentennamen. Ein* SONAR-Multiagentensystem *ist ein Tupel*

$$MAS = (\mathcal{A}_\mu, \mu),$$

wobei gilt:

- $\mathcal{A}_\mu \subseteq \mathcal{A}$ *ist eine endliche Menge an Agentennamen.*
- $\mu : \mathcal{A}_\mu \to \mathcal{S}$ *weist jedem Agentennamen einen* SONAR-*Agenten zu.*

Die *MAS-Grundstruktur* besteht aus den vorhandenen Agenten:

$$\mathcal{S}_{MAS} := \{\mu(A) \mid A \in \mathcal{A}_\mu\} \tag{3.17}$$

Jeder SONAR-Agent $\mu(A) \in \mathcal{S}_{MAS}$ besitzt die Komponenten $\Delta_{\mu(A)}$, $\delta_{\mu(A)}$ usw. Für jeden Agenten $A \in \mathcal{A}_\mu$ identifizieren wir A mit $\mu(A)$ und schreiben im Folgenden kurz Δ_A statt $\Delta_{\mu(A)}$ und analog für die anderen Komponenten.

Sei ein Multiagentensystem gegeben. Wie bereits erwähnt, muss das durch A_0 generierte Team

$$G_0 := \gamma_{A_0}(R_0) = (K_G, R_G, D_G, \psi_G, A_G)$$

zu den Absichten der beteiligten Agenten des Teams passen. Wir definieren $G_{\Uparrow p}$ als das Subteam ist, das aus $G = ((P, T, F), R, D, \psi, A)$ entsteht, wenn man es auf Knoten $\uparrow p$ (d.h. auf Knoten, die p bzgl. F^* nachfolgen) einschränkt:

$$G_{\Uparrow p} = ((P', T', F \cap (\uparrow p)^2), R|_{P'}, D|_{T'}, \psi|_{T'}, A|_{T'}) \quad \text{mit} \quad P' = P \cap \uparrow p, T' = T \cap \uparrow p$$

Jedes Subteam $G_{0\Uparrow p}$ von G_0 muss dann identisch mit dem Team sein, das $A_G(p)$ generiert. Auerdem müssen die Teamattribute zu den individuellen Präfenzen der Agenten passen. Wir formalisieren nun die dazu notwendigen Bedingungen.

Definition 43 *Sei ein Multiagentensystem $(\mathcal{A}_\mu, \mu)$ und ein Team $G = (K_G, R_G, D_G, A_G)$ mit $K_G = (P, T, F)$ gegeben. Das Multiagentensystem* akzeptiert *G, wenn gilt*

1. *Die Teamattribute (Dienste, Rollen und Agenten) passen zu den individuellen Präfenzen der Agenten:*

$$\forall (p, t) \in F_G : \forall \Delta : \delta_{A(t)}(G)(\Delta)(p) = (D_G(t), \{(R_G(p'), A_G(p'^\bullet)) \mid p' \in t^\bullet\})$$

2. *Das von $A_G(t)$ generierte Team $\gamma_{A_G(t)}(R_G(p))$ ist mit dem Subteam $G_{\Uparrow p}$ identisch:*

$$\forall (p, t) \in F_G : \gamma_{A_G(t)}(R_G(p)) = G_{\Uparrow p}$$

Die Menge aller vom Multiagentensystem akzeptierten Teams bezeichnen wir mit $\mathcal{G}_{akz}(\mathcal{A}_\mu, \mu)$.

Aus dieser Definition folgt, dass Teamformation erstens nicht von den habituellen Präferenzen und der Planung der Agenten zu trennen ist und dass sie zweitens ein verteilter Prozess ist.

Ist keines dieser Teams (im Sinne von Definition 43) akzeptierbar, so muss die Planung einiger (oder aller) Agenten nachkorrigiert werden, solange bis mindestens ein akzeptables Team existiert. Erfüllt dann keines dieser Teams die Teamsteuerung, so muss die lokale Steuerung ψ verschärft werden, bis die globale Steuerung Korrektheit garantiert – im Extremfall solange, bis die Korrektheit schon lokal erzwungen wird.

3.5.2 Wohlgeformte Teamplanung

Die einzelnen Pläne $\pi_A(G)(D)(p)$ der Agenten $A = A_G(t)$ für $(p,t) \in F_G$ müssen miteinander konsistent sein, d.h. zu einem gemeinsamen Teamplan π_G der Gruppe G passen. Ein Teamplan π_G ist ein Prozess des Teamnetzes $D(G)$, der zum einen die Steuerbedingung $\hat{\psi}(G)$ des Teams erfüllt und zum anderen einen korrekten Ablauf von $D(G)$ beschreibt. Die Menge aller Teampläne einer Gruppe G ist:

$$\begin{aligned} \mathit{Teampläne}(G) := \{ \quad & \pi_G \in \mathit{Proc}(D(G)) \mid \forall D \in \mathcal{D} : \forall t \in {K_G}^\circ : \\ & \pi_{A_G(t)}(G)(D)(p(t)) = \pi_G \wedge \pi_G \models \hat{\psi}_G(t) \wedge \\ & \phi({\pi_G}^\circ) = M_f \wedge \pi(M_f) = m_f \,\} \end{aligned} \tag{3.18}$$

Hierbei meint $\pi(M_f)$ die Projektion der gefärbten Markierung M_f auf ungefärbte Marken.

Die lokale Wahl eines Agenten für seinen Plan ist durch die Selektionsfunktion π bestimmt, für die $\pi(G)(D)(p) \in \mathit{Proc}(D[R_G(p)])$ gefordert wird. Dies ist verträglich mit der der Eigenschaft des Teamplans π_G aus der Menge $\mathit{Proc}(D(G)))$ zu sein, denn nach Lemma 6 gilt die Inklusion:

$$\mathit{Proc}(D(G), \hat{\psi}(G)) \subseteq \mathit{Proc}(D(G)) \subseteq \mathit{Proc}(D[R(p(t))])$$

Definition 44 *Ein Multiagentensystem $(\mathcal{A}_\mu, \mu)$ ist* wohlgeformt*, wenn für jeden Agent $A \in \mathcal{A}_\mu$ folgendes gilt:*

1. *A kennt nur Teampartner, die in $\mathcal{A}_\mu$ enthalten sind:*

$$\bigcup \{\mathcal{A}_\Delta(\delta) \mid G \in \mathcal{G}, p \in P_G, \delta \in \Delta_A(G)(p)\} \subseteq \mathcal{A}_\mu$$

2. *A vefügt über die benötigten Fähigkeiten:*

$$\forall G \in \mathcal{G} : \forall p \in P_G : \forall \delta \in \Delta_A(G)(p) : R_\Delta(\delta) \subseteq \rho(D_\Delta(\delta))$$

3. *In jeder Gruppe existiert ein gemeinsamer Teamplan π_G:*

$$\forall G \in \mathcal{G} : \exists \pi_G \in \mathit{Teampläne}(G)$$

In wohlgeformten Multigensystemen vefügen die Agenten über die im Team notwendigen Fähigkeiten.

Lemma 7 *Akzeptiert ein wohlgeformtes Multigensystem $(\mathcal{A}, \mu)$ das Team G, dann vefügt jeder Agent über die Fähigkeiten, die für die Ausführung der durch das Team festegelegten Rollen $R_G(t^\bullet)$ des Dienstes $D_G(t)$ notwendig sind:*

$$\forall t \in T : R_G(t^\bullet) \subseteq \rho_{A_G(t)}(D_G(t))$$

Beweis: Da das Team akzeptiert ist, entsprechen die Teamattribute den Präferenzen: $D_\Delta(\delta)) = D_G(t)$ und $R_\Delta(\delta) = R_G(t^\bullet)$. In einem wohlgeformten Multiagentensystem vefügt jeder Agent A über die benötigten Fähigkeiten: $R_\Delta(\delta) \subseteq \rho_A(D_\Delta(\delta))$, und jeder Agent kann nach Definition 41 aufgrund der Eigenschaft von Δ nur solche Dienste wählen. Daraus folgt für $A = A_G(t)$ sofort $R_G(t^\bullet) \subseteq \rho_{A_G(t)}(D_G(t))$. q.e.d.

Durch diese Bedingung besitzen alle Agenten im Team einen Teamplan, der mit den Randbedingungen des Teams vereinbar ist. Der gemeinsame Teamplan π_G ist im allgemeinen Ergebnis eines verteilten Planungs- und Aushandlungsprozesse. Von praktischer Bedeutung ist hier die Problematik, die einzelnen individuellen Pläne $\pi_{A_G(t)}(G)(D)(p)$ in einem gemeinsamen Teamplan π_G – einem Kompromiß – zu integrieren.

Theorem 31 *Wenn $(\mathcal{A}_\mu, \mu)$ ein wohlgeformtes Multiagentensystem ist, dann generiert jedes Team G nur solche Teampläne $\pi_G \in Proc(D(G))$, die die Teamsteuerung $\hat{\psi}(G)$ erfüllen.*

Beweis: Da das Multiagentensystem wohlgeformt ist, existiert für jeden Gruppe ein gemeinsamer Teamplan π_G mit $\pi_A(G)(D)(p) = \pi_G$, der die Steuerbedingung erfüllt. Außerdem gilt $\pi_A(G)(D)(p) \in Proc(D[R_G(p)])$ nach Definition der Selektionsfunktion. Für alle Teampläne gilt:

$$\begin{array}{rcl} & & \forall t \in {K_G}^\circ : \pi_G \models \hat{\psi}_G(t) \wedge \pi_G \in Proc(D[R_G(p)]) \\ & \iff & \forall t \in {K_G}^\circ : \pi_G \in Proc(D_G[R_G(p(t))], \hat{\psi}_G(t)) \\ & \iff & \pi_G \in \bigcap_{t \in {K_G}^\circ} Proc(D_G[R_G(p(t))], \hat{\psi}_G(t)) \\ \text{mit Thm. 20} & \iff & \pi_G \in Proc(\|_{t \in {K_G}^\circ} D_G[R_G(p(t))], \bigwedge_{t \in {K_G}^\circ} \hat{\psi}_G(t)) \\ & \iff & \pi_G \in Proc(D(G), \hat{\psi}(G)) \end{array}$$

Als Teamplan muss π_G auch maximal sein, d.h. er muss den Dienst bis in die finale Markierung bringen: $\phi(\pi_G{}^\circ) = M_f$. q.e.d.

Besonders praktisch sind solche Dienstnetze $D(G)$ des Teams G, die sogar unter der Steuerung $\hat{\psi}(G)$ noch schwach-korrekt sind, denn dann kann jedes Prozessanfangsstück eines gemeinsamen Teamplans stets zu einem terminierenden Prozess vervollständigt werden, da ein korrektes Dienstnetz in jeder erreichbaren Markierung nach Definition korrekt terminieren kann (vgl. Def. 30):

$$\begin{array}{rl} \forall M \in RS(D \times \psi, M_0) : & \pi(M) \geq m_f \Longrightarrow \pi(M) = m_f \wedge \\ & \exists M' \in RS(D \times \psi, M) : \pi(M') = m_f \end{array}$$

Ist $(D(G) \times \hat{\psi}(G))$ ein schwach-korrektes Dienstnetz, dann ist es also nicht notwendig, den Teamplan vorab gemeinsam festzulegen, da ja keine Aktion der Teammitglieder in eine Verklemmung führen kann. Die Verhandlungsphase der Teamplanung zur Festlegung von π_G kann daher entfallen. Da manche Teampläne aber besser als andere sein können, kann eine Verhandlung dennoch sinnvoll sein.

Wir definieren ein Teamnetz G als wohlkoordiniert, wenn der von ihm generierte Dienst $D(G)$ unter der Steuerung $\hat{\psi}(G)$ die Möglichkeit zur korrekten Termination besitzt.

Definition 45 *Eine Gruppe G heißt* wohlkoordiniert*, wenn $(D(G) \times \hat{\psi}(G))$ ein schwach-korrektes Dienstnetz ist. Ein wohlgeformtes Multiagentensystem $(\mathcal{A}_\mu, \mu)$ heißt* wohlkoordiniert*, wenn alle $G \in \mathcal{G}$ wohlkoordiniert sind.*

Die Eigenschaft eines Multiagentensystems, wohlkoordiniert zu sein, hängt direkt von der Struktur der Menge aller Teams $\mathcal{G}$ ab. Es ist daher im allgemeine nicht möglich, hinreichende Kriterien für die Wohlgeformtheit zu entwickeln, ohne etwas über den Aufbau von $\mathcal{G}$ zu wissen. Wir betrachten daher im folgenden Kapitel Organisationen als Strukturen, die die Menge $\mathcal{G}$ erzeugt, um so Aussagen über die Wohlgeformtheit treffen zu können.

Zusammenfassung

In diesem Kapitel haben wir die Koordinierung in Agentensystemem betrachten. Wir haben das Modell der Rollen/Dienst-Netze (R/D-Netze) entwickelt, um die

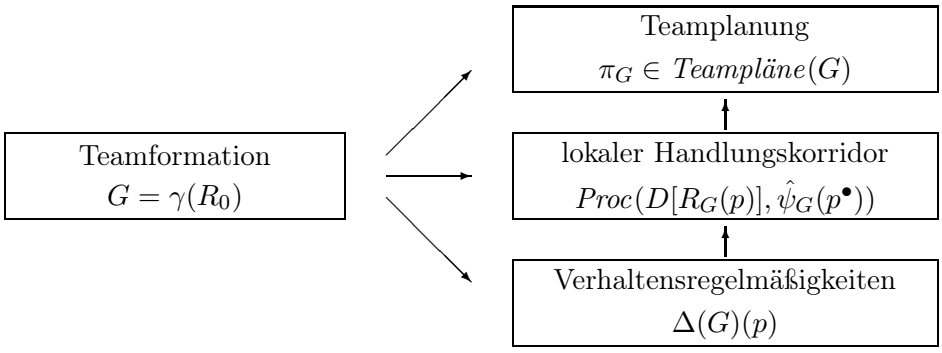


Abbildung 3.10: Der formale Teamplanungsprozess – vgl. Abb. 3.9

Interaktion in Teams zu beschreiben. Rollen/Dienst-Netze sind Petrinetze, deren Transitionen mit Dienstnetzen und Stellen mit Rollen verbunden sind. Teamnetze sind spezielle Rollen/Dienst-Netze, bei denen die Delegationsstruktur eindeutig festgelegt ist, was sich im Netz in der Eigenschaft wiederspiegelt, dass Teams Kausalnetze sind.

Rollen/Dienst-Netze sind spezielle Delegationsnetze, d.h. Netze, die genau eine Stelle im Vorbereich besitzen. Diese Eigfenschaft impliziert, dass die Erreichbarkeitsmengen von Delegationsnetzen linear sind und die Menge der Tätigkeitspfade kontextfrei. Diese Eigenschaft konnten wir ausnutzen, um die Bearbeitbarkeit von Aufgaben in Rollen/Dienst-Netzen effizient zu entscheiden.

Rollen/Dienst-Netze bilden die Grundlage für die Interaktion der SONAR-Agenten. Die Prozess-Semantik der Petrinetze impliziert, welche Teams sich formieren können und wie sich die Plangenerierung im Team vollzieht. Der formale Teamplanungsprozess ist – in Anlehunung an die Agentenarchitektur aus Abb. 3.9 – in Abbildung 3.10 zusammengefasst.

Der Formalismus erlaubte es uns darüberhinaus, Kriterien der Wohlgeformtheit für Teamstrukturen zu entwickeln. Für Rollen/Dienst-Netze bedeutet Wohlgeformtheit, dass die Dienstnetzanschriften der Transitionen zu den Rollenanschriften der Stellen passen. Neben diesem statischen Begriff der Wohlgeformtheit haben wir auch einen Begriff der Wohlgeformtheit des Teamplanungsprozesse entwickelt, der beschreibt, dass die lokalen Pläne der Agenten konsistent mit der Plangenerierung im Team sind.

Betrachten wir im folgenden Kapitel mit Organisationen einen Erzeugungsmechanismus für Teamnetze.

4 Formale Agentenorganisationen

Das SONAR-Multiagentensystem als Organisationsform besteht aus einer formalen Organisation und ihren Mitgliedern, d.h. den der Organisation zugeordneten Akteuren. Unter einer Organisation verstehen wir hier eine formalisierte Institution, die sich zudem durch die formale Regelung der Mitgliedschaft kennzeichnet. Der Organisationsbegriff ist also weiter gefasst als der betriebliche. In diesem Kapitel betrachten wir zunächst die formale Organisation. Kapitel 6 widmet sich dann der von Organisation und ihren Mitgliedern.

Eine Organisation wird in Hinblick auf ein Designziel konstruiert.[1] Das Organisationsdesign hat den Zweck, in Abhängigkeit der vorliegenden Aufgaben und der zur Verfügung stehenden Technologie die Organisationsstrukturen (Netzwerk der Organisationseinheiten, Rolle etc.) zu definieren. Die Wahl betrifft sowohl die Typisierung als auch die Interaktivitäten. Die Wahl der Organisationsstrukturen hat Auswirkungen darauf, wie effektiv oder effizient die Agenten Aufgaben – in Form von Aktivitäten – bearbeiten.

Zur organisationalen Kernaktivität zählen wir nur die Elemente der formalen Organisation, d.h. das Netzwerk, die Positionen, die Ressourcen, die Rollen und die Aktivitäten (vgl. dazu auch Abbildung 1.7). An dieser Stelle spielen Akteure – als System(binnen)umwelt – und ihre Fähigkeiten noch keine Rolle. Wir geben im folgenden ein Modell an, in dem sich alle Modellelemente (Positionen, Aktivitäten, Rollen, Netzwerk usw.) einfügen und zeigen, wie man diese zur Generierung von Multiagentensystemen nutzen kann.

4.1 Organisationspetrinetze und Positionen

Die Elemente der formalisierten Organisation sind die *Positionen*. Zentrales Konzept der Organisation ist das Organisationsnetzwerk, welches die Anordnung der Positionen definiert: Positionen stehen in Verbindung mit anderen Positionen. Neben den aus dem betrieblichen Kontext bekannten Organisationsstrukturen – wie der Hierarchie, dem Stab oder der Matrix – existiert eine Vielzahl von Strukturen, die im Kontext der Graphentheorie, des Hardwareentwurfs oder der verteilten Algorithmen studiert wurden. Beispiele sind hier die für Betriebe eher untypischen Stukturen des Kreises, des Hypercubes, des Butterfly-Netzwerkes usw.

Die Beziehungen zwischen den Postionen werden durch die *Verfahren* definiert. Verfahren regeln reflexiv die Mitgliedschaft von Akteuren und ihre Rollen in der Organisation, sie koordinieren die Interaktionen der Akteure miteinander, sie regeln die Modifikation von Verfahren. Durch Verfahren ist also sowohl die Statik als auch die die Dynamik einer Organisation geregelt.

Jeder formale Organisation lässt sich ein spezielles R/D Netz definieren: Jedes *Rolle* wird durch eine Stelle modelliert, jede *Tätigkeit* durch eine Transition. Jede Transition t besitzt genau eine Stelle im Vorbereich. Eine (t, p)-Kante bedeutet, dass die Tätigkeit t die Rolle p *nutzt*. Eine (p, t)-Kante bedeutet, dass die mit

[1] Von diesem Designziel sich die Organisation im Laufe ihrer Evolution entfernen.

p verbundenen Rollenaufgaben durch die Tätigkeit t *implementiert* werden. Eine Marke auf der Stelle p stellt einen Arbeitsauftrag dar, der von einer Tätigkeit $t \in {}^\bullet p$ generiert wurde und der von einer Tätigkeit $t \in p^\bullet$ bearbeitet werden muss.

Für Organisationen gruppieren wir Teilmengen von Stellen und Transitionen zu organisationalen Positionen: $O \subseteq P \cup T$. Auf diese Art und Weise erhalten wir eine Partionierung $\mathcal{O}$ der Knotenmenge $P \cup T$. Wir fordern, dass wenn eine Transition t zu einer Position O gehört, alle genutzten Rollen $p \in t^\bullet$ ebenfalls zur Position gehören: $t^\bullet \subseteq O$. Die nutzenden Rollen $p \in {}^\bullet t$ gehören dagegen nicht dazu: ${}^\bullet t \subseteq \bar{O}$, wobei $\bar{O}$ das Mengenkomplement bezüglich der Knotenmenge $P \cup T$ bezeichnet. Gehört p zu einer Position, so sind alle zugreifenden Dieste $t \in {}^\bullet p$ auch Element der Position, nicht aber die implementierenden Dienste $t \in p^\bullet$.

Eine Organisation ist ein R/D-Netz (vgl. Definition 36) mit einer Organisationsstruktur. Eine Organisation weist jedem Stelle p einer Position O im Organisationsnetz ein Rollenprofil $R(t)$ und jeder Transition t ein Dienstnetz $D(t)$ zu: Das Rollenprofil, die eine Positionsstelle p auszufüllen hat, ist durch die Anschrift $R(p)$ gegeben. Das zu einer Transition t zugehörige Dienstnetz ist durch die Anschrift $D(t)$ definiert. Abbildung 4.1 zeigt eine Organisationsnetz, bei dem die Dienstnetze und die Rollenprofile bereits als Anschriften enthalten sind. Analog zu Teams weisen wir jeder Transition t eine Prozessformel $\psi(t)$ zu, die eine Menge zugelassener Prozesse ihres Dienstnetzens $D(t)$ beschreibt.

Definition 46 *Sei $N = (P, T, F)$ ein P/T-Netz. Eine Partionierung $\mathcal{O}$ auf der Knotenmenge $P \cup T$ ist eine* Organisationsstruktur*, wenn folgende Zusammenhangsbedingung erfüllt ist:*

$$\begin{array}{rl} \forall O \in \mathcal{O}: & (\forall p \in O \cap P: \quad {}^\bullet p \subseteq O \quad \wedge \quad p^\bullet \subseteq \bar{O}) \wedge \\ & (\forall t \in O \cap T: \quad {}^\bullet t \subseteq \bar{O} \quad \wedge \quad t^\bullet \subseteq O) \end{array}$$

Ein Element $O \in \mathcal{O}$ heißt Position *der Organisation.*

Ein Organisationsnetz $(N, \mathcal{O})$ *besteht aus einer Organisationsstruktur $\mathcal{O}$ zum Netz N.*

Sei eine Rollenmenge $\mathcal{R}$ und eine Dienstklasse $(\mathcal{D}, D^{\mathcal{D}})$ gegeben. Eine Organisation

$$Org = (N, \mathcal{O}, R, D),$$

besteht aus dem Organisationsnetz $(N, \mathcal{O})$ und dem R/D-Netz (N, R, D).

Eine Organisation $(N, \mathcal{O}, R, D)$ heißt wohlgeformt*, wenn das R/D-Netz (N, R, D) dies ist.*

Eine koordinierende Organisation *ist das Paar (Org, ψ), wobei $\psi : T \to PF_{\mathcal{D}}$ die zugelassenen Prozesse der Organisation Org definiert, wobei $\forall t \in T : Proc(D(t), \psi(t)) \neq \emptyset$ gelten muss.*

Koordinierende Organisationen notieren wir auch in der Form $Org = (N, \mathcal{O}, R, D, \psi)$. Definieren wir $\psi(t) = \textsc{True}$, so erkennen wir, dass koordinierende Organisationen eine konservartive Erweiterung der Organisation darstellen. Besteht keine Gefahr der Verwechslung sprechen wir auch im Falle koordinierender Organisationen nur von Organisationen.

Da $\mathcal{O}$ eine Partition der Menge $P \cup T$ ist, existiert zu jeder Position $p \in P$ genau eine Position $O \in \mathcal{O}$, die wir mit $O(p)$ bezeichnen. Genauso existiert zu jeder Tätigkeit $t \in T$ genau eine Position $O \in \mathcal{O}$, die wir mit $O(t)$ bezeichnen.

Theorem 32 *Sei $\mathcal{O}$ eine Organisationsstruktur.*

- *Jede Position $O \in \mathcal{O}$, die keine Tätigkeiten besitzt, wird von keiner Position genutzt:* $\forall O \in \mathcal{O} : O \subseteq P \Longrightarrow {}^\bullet O = \emptyset$.
- *Jede Position $O \in \mathcal{O}$, die keine Rollenprofile enthält, kann keine anderen Position nutzen:* $\forall O \in \mathcal{O} : O \subseteq T \Longrightarrow O^\bullet = \emptyset$.

Beweis: Sei $O \subseteq P$ und $O \in \mathcal{O}$. Aus der Definition folgt für alle $p \in O \cap P$, dass ${}^\bullet p \subseteq O$ gelten muss. Da aber stets ${}^\bullet p \subseteq T$ gilt, folgt daher ${}^\bullet p = \emptyset$. Analog für $O \subseteq T$. q.e.d.

Eine Position $O \in \mathcal{O}$ mit $O \subseteq P$ heißt *initial*, denn diese Position startet eine Tätigkeit, ohne dass eine andere Positionen auf sie zurückgriffe. Eine Position $O \in \mathcal{O}$ mit $O \subseteq T$ heißt *terminal*, denn diese Position greift auf keine anderen Positionen zurück. Eine Position $O \in \mathcal{O}$ mit $O \subseteq P$ oder $O \subseteq T$ heißt *elementare* Position. Eine Position $O \in \mathcal{O}$ mit $O \cap P \neq \emptyset$ und $O \cap T \neq \emptyset$ modelliert dagegen eine *komplexe* Position, die auf andere Positionen zurückgreift.

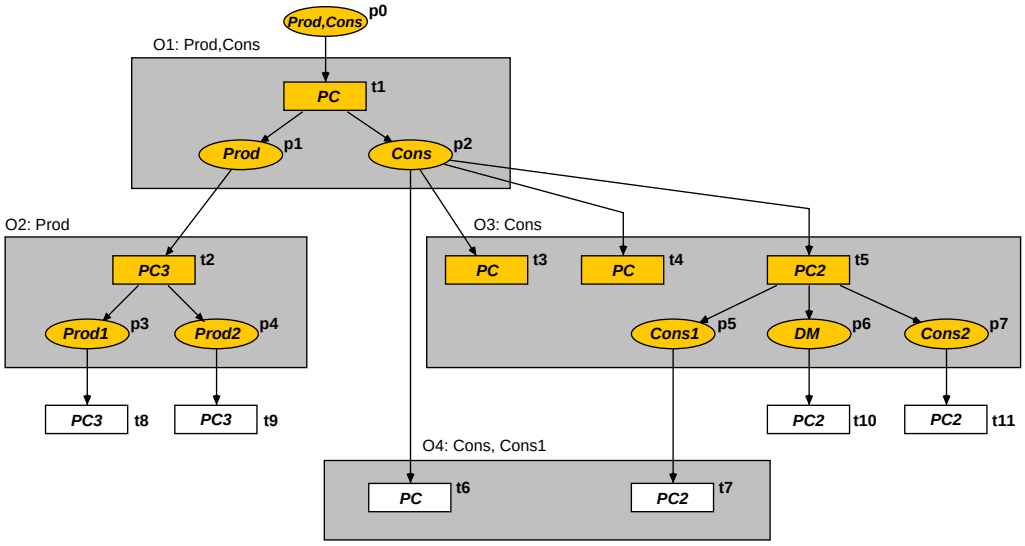


Abbildung 4.1: Organisationsnetz

Beispiel Abbildung 4.1 zeigt ein Organisationsnetz. Das der Organisation zugrundeliegende R/D-Netz ist mit dem in Abbildung 3.4 dargestellten Netz identisch. Die Menge der Positionen ist durch die Organisationstruktur $\mathcal{O} = \{O_0, \ldots, O_8\}$ gegeben, wobei die Elemente der Partition gegeben sind durch:

$$\begin{array}{lllllllll} O_0 & = & \{p_0\} & O_3 & = & \{t_3, t_4, t_5, p_5, p_6, p_7\} & O_6 & = & \{t_9\} \\ O_1 & = & \{t_1, p_1, p_2\} & O_4 & = & \{t_6, t_7\} & O_7 & = & \{t_{10}\} \\ O_2 & = & \{t_2, p_3, p_4\} & O_5 & = & \{t_8\} & O_8 & = & \{t_{11}\} \end{array}$$

Positionen, die mehr als einen Knoten umfassen, sind durch graue Kästen dargestellt. Die Positionen $O_1, \ldots, O_4$ sind komplex. Die Position O_0 ist eine reine Stellenmenge, stellt also eine Aktivierungsbedingung dar. Die Positionen $O_5, \ldots, O_8$ sind reine Transitionsmengen, sind also elementare Positionen. In O_3 können die Tätigkeiten t_3, t_4 und t_5 alternativ genutzt werden, um die Aufgabe p_2 zu realisieren.

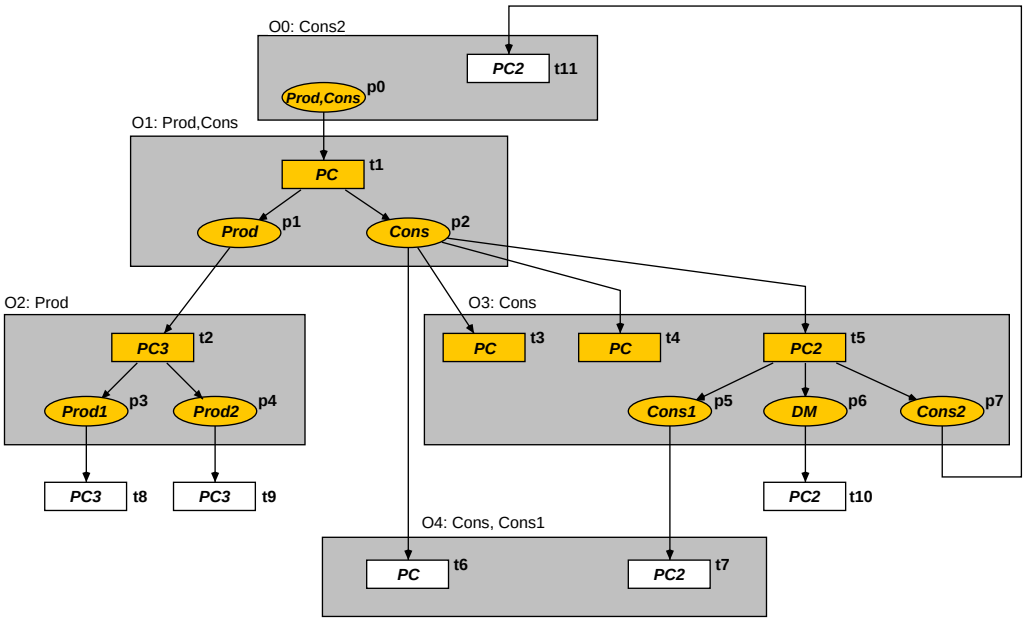


Abbildung 4.2: Alternatives Organisationsnetz

Abbildung 4.1 ist dabei natürlich weder das einzig mögliche, noch unter den zum R/D-Netz aus Abbildung 3.4 passenden Organisationsnetzen ein besonders ausgezeichnetes. Eine ebenso mögliche Alternative, die sogar Zyklen enthält, ist in Abbildung 4.2 dargestellt.

Man beachte, dass die Zusammenhangsbedingung, die eine Organisationsstruktur $\mathcal{O}$ zu erfüllen hat, verhindert, dass sich eine Position an der Bearbeitung der von ihr generierten Teilaufgaben beteiligen kann. So ist beispielsweise in Abbildung 4.1 die Transition t_{10}, welche die Rolle *DecisionMaker*(DM) implementiert, nicht zur Position O_3. Dies stellt aber zentrale keine Einschränkung dar, da die Positionen im folgenden (vgl. Abschnitt 4.3) weiter zu Positionsagenten gruppiert werden. Für die Organisation in Abbildung 4.1 wäre es naheliegend, die Positionen O_3 und O_7 zusammenzufassen, damit der delegierende Agent mit dem die Rolle *DecisionMaker* implementierden identisch ist. ◇

Lemma 8 *Sei $N = (P, T, F)$ ein Netz. Besitzt N Schlingen, dann ist es nicht möglich, eine Organisationsstruktur anzugeben Ist N schlingenfrei, dann existiert eine Organisationsstruktur $\mathcal{O}$.*

Beweis: Für ein Netz N mit Schlingen ist es nicht möglich, eine Organisationsstruktur anzugeben, da jeder Zyklus $pFtFp$ die Zusammenhangsbedingung verletzt.

Zu jedem schlingenfreien Netz können wir eine Organisationsstruktur definieren: Zu jeder Transition t betrachten wir die Menge der Knoten p_i, t_i, die wir zu einer Position O hinzunehmen müssen, um die Zusammenhangsbedingung nicht zu verletzen:

$$\{p_i, t_i \mid tFp_1F^{-1}t_1Fp_2F^{-1}t_2Fp_3 \ldots\}$$

Analog setzen wir für einen Platz p die Knoten $\{p_i, t_i \mid pF^{-1}t_1Fp_1F^{-1}t_2 \ldots\}$ in

Beziehung. Dies können wir durch eine einzige Relation ausdrücken.

$$\sim \;:=\; \Big(\big((F \circ F^{-1})^* \cup (F \circ F^{-1})^* \circ F\big) \;\cap\; \big(T \times (P \cup T)\big)\Big) \cup \Big(\big((F^{-1} \circ F)^* \cup (F^{-1} \circ F)^* \circ F^{-1}\big) \;\cap\; \big(P \times (P \cup T)\big)\Big)$$

Man sieht leicht, dass $\sim$ eine Äquivalenzrelation auf den Netzknoten darstellt, also eine Partition $\mathcal{O}$ erzeugt. q.e.d.

4.1.1 Typische Organisationformen

Ein Markt ist Abbildung 4.3 dargestellt, ein Koalition in Abbildung 4.4 und eine virtuelle Organisation in Abbildung 4.5. Die Netze heben jeweils schön die spezifischen Eigenschaften der Organisationsformen heraus.

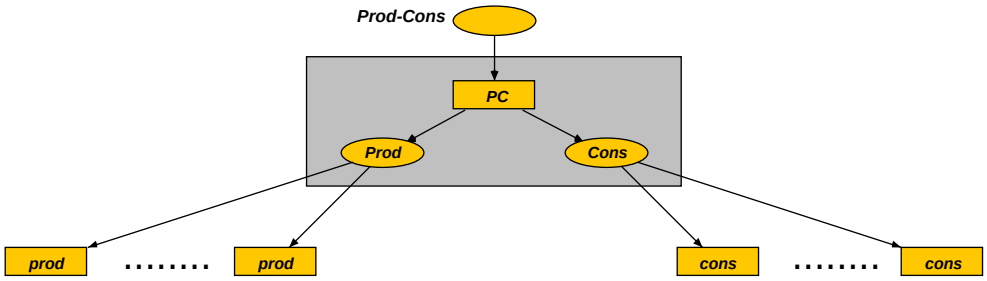


Abbildung 4.3: Ein Markt als Organisationsnetz

Der Markt in Abb. 4.3 zeichnet sich durch seine flache Struktur aus, es findet also kaum iterierte Delegation statt. Außerdem strukturiert der Markt genau eine Rollenbeziehung, nämlich die von Erzeuger und Verbraucher. Dafür ist der Verzweigungsgrad sehr hoch, da es viele Marktteilnehmer gibt. Alle entstehenden Teams sind also – gemessen an der Tiefe der Graphenstruktur – sehr flach. Die Anzahl der generierbaren Team, die ja mit dem Alternativgrad wächst, ist dagegen sehr groß.

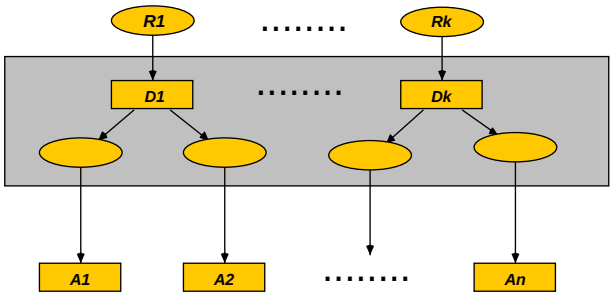


Abbildung 4.4: Eine Koalition als Organisationsnetz

Die Koalition in Abb. 4.4 zeichnet sich dagegen durch eine geringere Anzahl an Mitgliedern, aber eine höhere Anzahl an Aktivitäten aus. Das Hauptaugenmerk liegt auf der Bündelung der verschiedenen Aktivitäten zu einem kohärentem Ganzen. Entsprechend hoch ist die Anzahl der Aufgaben und Rollen, die die Organisation bearbeiten kann. Für jede einzelne der Aufgaben ist die Anzahl der Bearbeitungsvarianten dagegen eher niedrig.

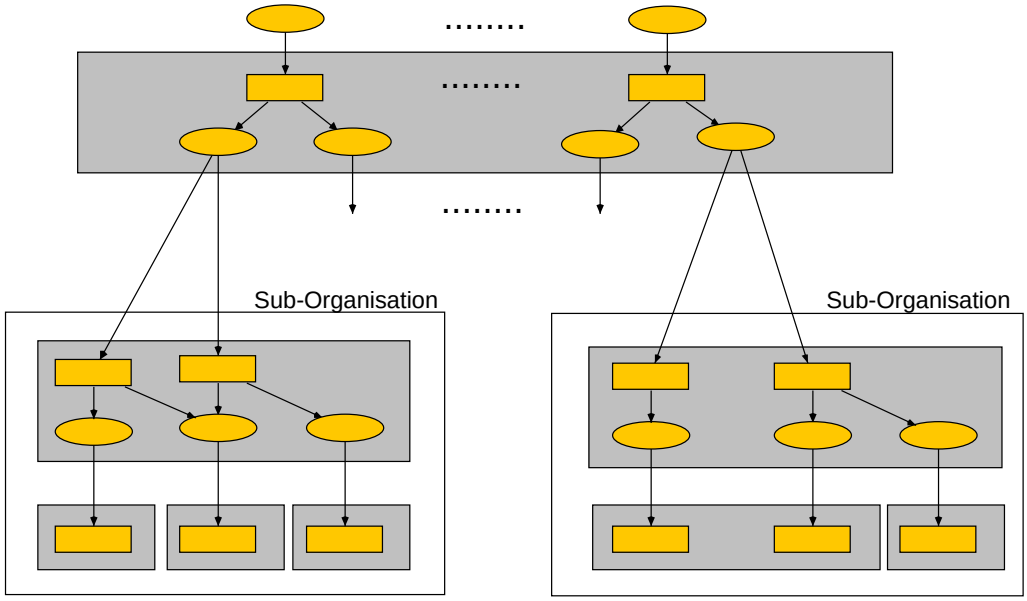


Abbildung 4.5: Eine virtuelle Organisation als Organisationsnetz

Die virtuelle Organisation aus Abb. 4.5 erweitert diesen Aspekt der Bündelung um den Aspekt der Tiefenstruktur. Dieser zeigt sich durch die größere Delegationstiefe der Aufgaben, die sich ergibt, weil sich in einer virtuelle Organisation bereits vorstrukturierte Suborganisation mit vielen Positionen, die weitläufiger verbunden sind, zusammenschließen.

4.1.2 Organisationsverbünde

Im folgenden wollen wir verschachtelte Organisationen betrachten. Eine Hierarchie von Organisationen entsteht, wenn eine Position durch eine komplette Organisation verfeinert wird. Eine solche Verfeinerung von Organisationen können wir äquivalent als Vergöberung ausdrücken, indem wir umgekehrt eine Menge $\mathcal{Q} \subseteq \mathcal{O}$ von Positionen zu einer Position $O_{\mathcal{Q}}$, die aus einer einzigen Transition besteht, vergröbern. Diese neue Transition bezeichnen wir mit $t_{\mathcal{Q}}$.

Damit das Ergebnis der Konstruktion wieder ein R/D-Netz ist, muss die neue Transition $t_{(P_{\mathcal{Q}}{}^{\bullet} \setminus \bigcup \mathcal{Q})}$ wiederum genau eine Stelle im Vorbereich besitzt. Daher ist es wichtig, dass die Tätigkeiten am Rand mit genau einer Stelle verbunden sind: $|{}^{\bullet}T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}| = 1$.

Analog können wir anhand der Positionsmenge $\mathcal{Q}$ eine Suborganisations definieren, indem wir uns auf alle Netzknoten aus $\mathcal{Q}$ sowie die Im Rand ${}^{\bullet}T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}$ davor liegenden Stellen beschränken. Letztere benötigen wir, um der Suborganisation von außen Aufgaben zuweisen zu können.

Definition 47 *Sei* $Org = (N, \mathcal{O}, R, D)$ *eine Organisation. Die Position-Menge* $\mathcal{Q} \subseteq \mathcal{O}$ *heißt* semi-abgeschlossen, *falls mit* $P_{\mathcal{Q}} := P \cap \bigcup \mathcal{Q}$ *und* $T_{\mathcal{Q}} := T \cap \bigcup \mathcal{Q}$ *gilt:*

1. *Alle Rollenprofile werden nur lokal genutzt:* $|P_{\mathcal{Q}}{}^{\bullet} \setminus \bigcup \mathcal{Q}| = 0$
2. *Die Tätigkeiten am Rand werden exklusiv genutzt:* $|{}^{\bullet}T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}| = 1$

Sei $\mathcal{Q} \subseteq \mathcal{O}$ semi-abgeschlossen, dann ist die Suborganisation *von Org bezüglich $\mathcal{Q}$ folgendermaßen definiert:*

$$Org_{\downarrow\mathcal{Q}} = (N_{\downarrow\mathcal{Q}}, \mathcal{O}_{\downarrow\mathcal{Q}}, R_{\downarrow\mathcal{Q}}, D_{\downarrow\mathcal{Q}})$$

mit $N_{\downarrow\mathcal{Q}} = (P_{\downarrow\mathcal{Q}}, T_{\downarrow\mathcal{Q}}, F_{\downarrow\mathcal{Q}})$ und:

$$\begin{aligned}
P_{\downarrow\mathcal{Q}} &= P_{\mathcal{Q}} \cup {}^\bullet T_{\mathcal{Q}} \\
T_{\downarrow\mathcal{Q}} &= T_{\mathcal{Q}} \\
F_{\downarrow\mathcal{Q}} &= \Big(F \cap ((T_{\downarrow\mathcal{Q}} \times P_{\downarrow\mathcal{Q}}) \cup (P_{\downarrow\mathcal{Q}} \times T_{\downarrow\mathcal{Q}}))\Big) \\
\mathcal{O}_{\downarrow\mathcal{Q}} &= \mathcal{Q} \cup \Big\{ {}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q} \Big\} \\
R_{\downarrow\mathcal{Q}} &= R|_{P_{\downarrow\mathcal{Q}}} \\
D_{\downarrow\mathcal{Q}} &= D|_{T_{\downarrow\mathcal{Q}}}
\end{aligned}$$

Sei $\mathcal{Q} \subseteq \mathcal{O}$ semi-abgeschlossen, dann ist die Vergröberung *von Org bezüglich $\mathcal{Q}$ die Organisation*

$$Org_{/\mathcal{Q}} = (N_{/\mathcal{Q}}, \mathcal{O}_{/\mathcal{Q}}, R_{/\mathcal{Q}}, D_{/\mathcal{Q}})$$

mit $N_{/\mathcal{Q}} = (P_{/\mathcal{Q}}, T_{/\mathcal{Q}}, F_{/\mathcal{Q}})$ und:

$$\begin{aligned}
P_{/\mathcal{Q}} &= P \setminus \bigcup \mathcal{Q} \\
T_{/\mathcal{Q}} &= \Big(T \setminus \bigcup \mathcal{Q}\Big) \uplus \{t_{\mathcal{Q}}\} \\
F_{/\mathcal{Q}} &= \Big(F \cap \big((T_{/\mathcal{Q}} \times P_{/\mathcal{Q}}) \cup (P_{/\mathcal{Q}} \times T_{/\mathcal{Q}})\big)\Big) \cup \Big(({}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}) \times \{t_{\mathcal{Q}}\}\Big) \\
\mathcal{O}_{/\mathcal{Q}} &= (\mathcal{O} \setminus \mathcal{Q}) \cup \{O_{\mathcal{Q}}\} \quad \textit{mit} \quad O_{\mathcal{Q}} = \{t_{\mathcal{Q}}\} \\
R_{/\mathcal{Q}} &= R|_{P_{/\mathcal{Q}}} \\
D_{/\mathcal{Q}} &= D|_{T_{/\mathcal{Q}}} \cup \{(t_{\mathcal{Q}}, D^{\mathcal{D}}(R(p(t_{\mathcal{Q}}))))\}
\end{aligned}$$

Wenn $Org_{/\mathcal{Q}}$ eine Vergröberung von Org, dann nennen wir Org eine Verfeinerung *von $Org_{/\mathcal{Q}}$ durch $\mathcal{Q}$.*

Die semi-abgeschlosse Position-Menge $\mathcal{Q} = \{O_3, O_4, O_5, O_6\}$ aus Abbildung 4.6 kann zu einer einzigen Position $O_{\mathcal{Q}} = \{t_{\mathcal{Q}}\}$ vergröbert werden. Diese enthält nur die Transition $t_{\mathcal{Q}}$ und realisiert die Rolle $R(O_{\mathcal{Q}}) = \{Cons\}$.

Sei *Org* eine Organisation. Wird eine semi-abgeschlosse Position-Menge $\mathcal{Q}$ zu einer einzigen Position vergröbert, so sind Aufgaben von der resultierenden Organisation $Org_{/\mathcal{Q}}$ bearbeitbar, wenn es die verfeinerte ist.

Analog ergibt die Verfeinerung eines sicher-berarbeitbares Netz $Org_{/\mathcal{Q}}$ durch ein semi-abgeschlossenes, sicher-berarbeitbares Netz in Form der Suborganisation $Org_{\downarrow\mathcal{Q}}$ wiederum ein sicher-berarbeitbares Netz, nämlich *Org*.

Theorem 33 *Wenn $Org = (N, \mathcal{O}, R, D)$ eine Organisation und $\mathcal{Q} \subseteq \mathcal{O}$ eine semi-abgeschlosse Positionsmenge ist, dann ist die Vergröberung $Org_{/\mathcal{Q}}$ und die Suborganisation $Org_{\downarrow\mathcal{Q}}$ genau dann sicher bearbeitbar, wenn es Org ist.*

Beweis: ($\Leftarrow$) Da $\mathcal{Q}$ semi-abgeschlossen ist, gilt $|{}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}| = 1$, d.h. der Rand wird exklusiv genutzt. Sei $p_{\mathcal{Q}}$ das Element im Rand ${}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}$. Sei *Org* sicher bearbeitbar, dann gilt für alle $m' \in RS(m)$ auch $\mathbf{0} \in RS(m')$. Dies gilt insbesondere für $m = {}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q} = \{p_{\mathcal{Q}}\}$. Alle Stellen in $Org_{/\mathcal{Q}}$ sind dann auch bearbeitbar, denn

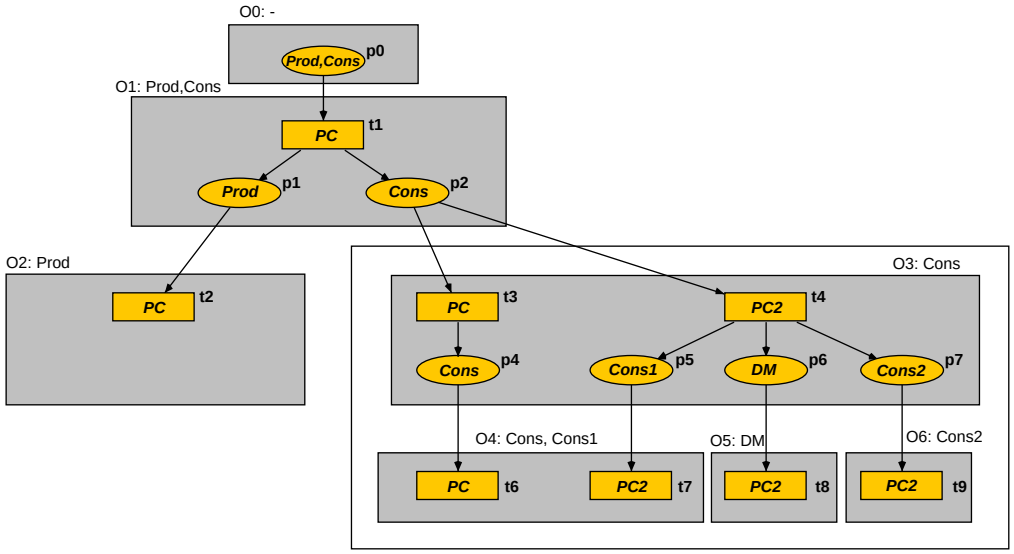


Abbildung 4.6: Organisation des Producer/Consumer-Systems

alle Stellen in $P_{/\mathcal{Q}}$ sind bearbeitbar. Die Stelle $p_{\mathcal{Q}}$ wird durch die Transition $t_{\mathcal{Q}}$ geleert. Die anderen Stellen nutzen nur Transitionen aus $T_{/\mathcal{Q}}$.

Wenn Org sicher bearbeitbar, dann sind auch alle Stelle p markierbar, also auch alle aus $\mathcal{Q}$. Da $\mathcal{Q}$ semi-abgeschlossen ist, führt aber keine Kante mehr aus der Suborganisation heraus, so dass die Marken innerhalb von $\mathcal{Q}$ bearbeitet werden

($\Rightarrow$) Seien nun die Vergröberung $Org_{/\mathcal{Q}}$ und die Suborganisation $Org_{\downarrow\mathcal{Q}}$ sicher bearbeitbar.

Wenn $Org_{/\mathcal{Q}}$ sicher bearbeitbar ist, dann gilt für alle $m' \in RS(m)$ auch $\mathbf{0} \in RS(m')$. Wir zeigen, dass dann auch die sichere Bearbeitbarkeit in Org gilt.

Jede Transition in $Org_{/\mathcal{Q}}$, kann in Org direkt simuliert werden, da die Transitionen aus $Org_{/\mathcal{Q}}$ auch in Org enthalten sind.

Sei $p_{\mathcal{Q}}$ das Element in ${}^\bullet T_{\mathcal{Q}} \setminus \bigcup \mathcal{Q}$. Jede Marke der Stelle $p_{\mathcal{Q}}$ in $Org_{\downarrow\mathcal{Q}}$ kann in $Org_{\downarrow\mathcal{Q}}$ bearbeitet werden. Daher kann jedes Schalten von $t_{\mathcal{Q}}$, welches die Stelle $p_{\mathcal{Q}}$ in $Org_{/\mathcal{Q}}$ leert, durch eine Schaltfolge in $Org_{\downarrow\mathcal{Q}}$ und daher auch in Org simuliert werden. q.e.d.

Diese strukturierte Form der Verfeinerung ist ein mächtiges Hilfsmitttel für die Strukturierung eines Multiagentensystems. Wir können den Mechanismus jedoch nicht nur zum Organisationsdesign nutzen, sondern auch zudem noch flexibel als Transformation, die wir zur Laufzeit anwenden.

4.2 Koordinierung der Organisationsprozesse

Um nebenläufige Ereignisse nicht unnötig durch eine sequentielle Notation zu unterscheiden, betrachten wir die partiell geordneten Prozesse von Organisationsnetzen. Abbildung 4.7 zeigt einen Prozess der Organisation aus Abbildung 4.1 in der Markierung $m = p_0$.

Ist ein Organisationsnetz N gegeben, so lassen sich anhand eines gegebenen Team-Prozesses (K, ϕ) die Dienst- und Rollenabbildung D und R leicht angeben, indem

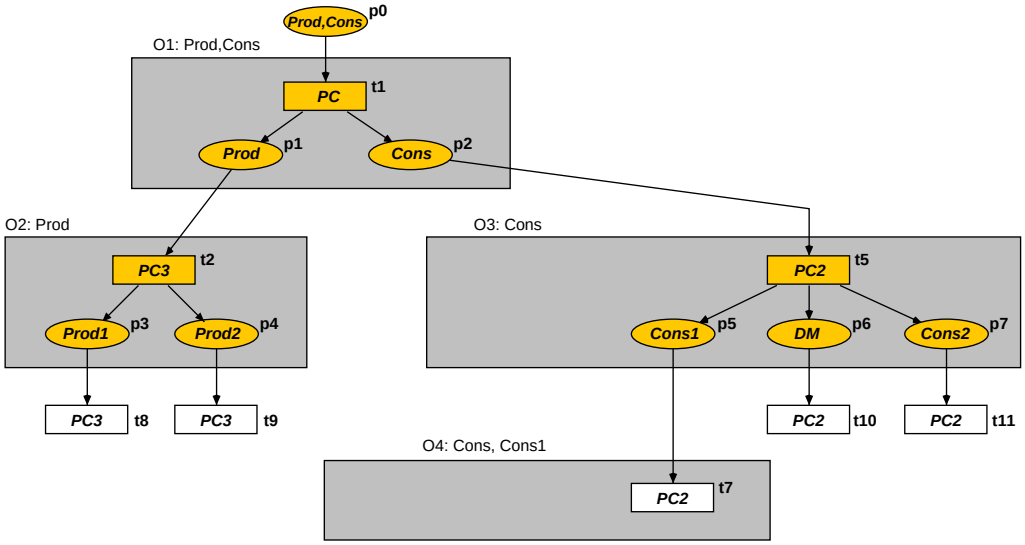


Abbildung 4.7: Prozess eines Organisationsnetzes

man für alle $b \in B$ und $e \in E$ definiert:

$$R_G(b) = R(\phi(b)) \quad \text{und} \quad D_G(e) = D(\phi(b))$$

Jeder Prozess der Organisation erzeugt ein koordiniertes Team.

Theorem 34 *Sei $Org = (N, \mathcal{O}, R, D)$ eine Organisation und $R \in \mathcal{R}$ eine Rolle. Dann generiert jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ zu einem $p \in P_{Org}(R)$ das R/D-Netz:*

$$G_{Org}(K, \phi) := (K, (R \circ \phi), (D \circ \phi))$$

Sei (Org, ψ) eine koordinierende Organisation und $R \in \mathcal{R}$ eine Rolle. Für jedes $p \in P_{Org}(R)$ erzeugt jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ des Organisationsnetzes das koordinierende Team:

$$G_{(Org,\psi)}(K, \phi) := (K, (R \circ \phi), (D \circ \phi), (\psi \circ \phi_{|E}))$$

Beweis: Die Abbildung $(R \circ \phi)$ und $(D \circ \phi)$ sind offensichtlich Rollen- und Dienstzuweisungen. Damit ist $(K, (R \circ \phi), (D \circ \phi))$ ein ein R/D-Netz. Außerdem stellt $\psi \circ \phi_{|E}$ eine Steuerung dar.

Offensichtlich ist K ein Kausalnetz. Es besitzt genau einen Platz als minimalen Knoten besitzt, da der Prozess zu $(N, \{p\})$ generiert wurde. Nach Theorem 27 gilt zudem $K^\circ \subseteq E$. Damit ist $(K, (R \circ \phi), (D \circ \phi), (\psi \circ \phi_{|E}))$ ein Teamnetz. q.e.d.

Die lokale Steuerung $\psi_K = (\psi \circ \phi_{|E})$ erweitert sich nach Definition 38 zu der globalen Steuerung $\hat{\psi}$.

Das folgende Theorem zeigt, dass die an eine koordinierende Organisation gestellte Bedingung $\forall t \in T : Proc(D(t), \psi(t)) \neq \emptyset$ konstruktiv entscheidbar ist.

Theorem 35 *Sei Org eine wohlgeformte Organisation, dann ist die Menge aller Prozesse*

$$\mathcal{K}^{pg}(Org) := \bigcup_{p \in P} \mathcal{K}^{pg}(N, \{p\})$$

eine endliche, konstruktive Menge.

Beweis: Dies folgt aus der Tatsache, dass jedes Prozessnetz (K, ϕ), die folgende Eigenschaft besitzt:

$$(\phi(b_1) = \phi(b_2)) \Longrightarrow (b_1 = b_2)$$

Denn wenn $\phi(b_1) = \phi(b_2)$ für zwei Stellen b_1 und b_2 wäre, dann wäre $R(\phi(b_1)) = R(\phi(b_2))$ in dem Teamnetz $G_{Org}(K, \phi)$ und nach Theorem 25 folgt dann $b_1 = b_2$.

Im jedem Prozess besitzt jede Stelle somit höchstens ein Urbild. Die Menge aller Prozesse, mit dieser Eigenschaft ist offensichtlich endlich und auch konstruktiv. q.e.d.

Jede Organisation ist somit ein Erzeuger einer Menge von Teamnetzen:

$$\mathcal{G}(Org) := \{G_{Org}(K, \phi) \mid (K, \phi) \in \mathcal{K}^{pg}(Org)\} \tag{4.1}$$

Damit können wir ein Multiagentensystem als organisiert definieren, wenn die Menge seiner Teamnetze $\mathcal{G}$ durch eine wohlgeformte Organisation *Org* erzeugt wird: $\mathcal{G} = \mathcal{G}(Org)$ und alle Teamnetze im Sinne von Definition 43 vom Multiagentensystem akzeptiert werden: $\mathcal{G} = \mathcal{G}_{akz}(\mathcal{A}_\mu, \mu)$.

Definition 48 *Das zur Teammenge $\mathcal{G}$ gebildete Multiagentensystem $(\mathcal{A}_\mu, \mu)$ heißt* organisiert*, wenn es eine wohlgeformte Organisation Org gibt, so dass gilt:*

$$\mathcal{G} = \mathcal{G}_{akz}(\mathcal{A}_\mu, \mu) = \mathcal{G}(Org)$$

4.2.1 Koordinierung: Realisierbarkeit der Steuerung

Ein Organisationsprozess (K, ϕ) kann die Steuerung *realisieren*, wenn es für jedes Ereignis e möglich ist, einen Prozess π des Dienstes $D(e)$ zu finden, der die globale Steuerung $\hat{\psi}_K(e)$ erfüllt:

$$(K, \phi) \models \hat{\psi}_K :\Longleftrightarrow \forall e \in E_K : \exists \pi \in Proc(D(e)[R(b(e))]) : \pi \models \hat{\psi}_K(e) \tag{4.2}$$

Für die Wohlgeformtheit einer koordinierender Organisation fordern wir, dass jeder Organisationsprozess die globale Steuerung realisieren kann.

Die Dienstnetze $D(t)$ und Rollenprofile $R(p)$ müssen zur Struktur des Organisationsnetzes passen, d.h wohlgeformt sein. Hier lässt sich die Begriffsbildung aus dem Bereich der Teams verwenden nach Definition 37 und 38.

Definition 49 *Eine Steuerung ψ heißt* wohlgeformt*, wenn für alle Organisationsprozesse mindestens ein Dienstprozess existiert, der die globale Steuerung $\hat{\psi}$ erfüllt:*

$$\forall R \in \mathcal{R} : \forall p \in P_{Org}(R) : \forall (K, \phi) \in \mathcal{K}^{pg}(N, \{p\}) : (K, \phi) \models \hat{\psi}_K$$

Eine koordinierende Organisation (Org, ψ) heißt wohlgeformt*, wenn Org und ψ wohlgeformt sind.*

Wohlgeformte Organisationen erzeugen nur wohlgeformte Teams.

Theorem 36 *Ist die Organisation $Org = (N, \mathcal{O}, R, D)$ wohlgeformt, so ist für alle Prozesse $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ auch $G_{Org}(K, \phi)$ ein wohlgeformtes R/D-Netz.*

Ist die koordinierende Organisation (Org, ψ) wohlgeformt, dann ist für alle Prozesse $(K, \phi) \in \mathcal{K}^{pg}(N, m)$ auch $G_{(Org,\psi)}(K, \phi)$ ein wohlgeformtes koordinierendes Team.

Beweis: Da *Org* wohlgeformt ist und Wohlgeformtheit eine Eigenschaft ist, die sich nur auf die Lokalität bezieht, gilt sie auch in den Entfaltungen von N, d.h. für alle Prozesse $(K,\phi) \in \mathcal{K}^{pg}(N,m)$.

Da (Org,ψ) wohlgeformt ist, gilt $(K,\phi) \models \hat{\psi}_K(e)$ für alle $R \in \mathcal{R}$, $p \in P_{Org}(R)$ und $(K,\phi) \in \mathcal{K}^{pg}(N,\{p\})$.

Dabei gilt $(K,\phi) \models \hat{\psi}_K(e)$ nach (4.2) genau dann, wenn für alle $e \in E_K$ ein Prozess π mit $\pi \models \hat{\psi}_K(e)$ existiert, d.h. das erzeugte Team $G_{(Org,\psi)}(K,\phi) = (K,(R\circ\phi),(D\circ\phi),(\psi\circ\phi_{|E}))$ ist wohlgeformt. q.e.d.

Betrachten wir zunächst eine hinreichende Charakterisierung der Wohlgeformtheit. Die maximal strenge Steuerung der Organisation (Org,ψ) ist definiert als

$$\xi^{\max}_{Org} := \bigwedge_{t\in T_N} \hat{\psi}(t) \tag{4.3}$$

Offensichtlich gilt für die maximal strenge Steuerung:

$$\forall t \in T_N : \xi^{\max}_{Org} \Longrightarrow \hat{\psi}(t) \Longrightarrow \psi(t)$$

Ein besonderer Fall liegt vor, wenn für alle Dienstnetze $D(t)$ ein Dienstprozess π existiert, der $\xi^{\max}_{Org}$ erfüllt. In diesem Fall realisieren alle Organisationsprozesse die Steuerung, woraus folgt, dass die Organisation wohlkoordiniert ist.

Theorem 37 *Sei $Org = (N,\mathcal{O},R,D,\psi)$ eine wohlgeformte, koordinierende Organisation. Existiert zu jedem $t \in T_N$ ein $\pi \in Proc(D(t),\xi^{\max}_{Org})$, dann ist (Org,ψ) wohlgeformt.*

Beweis: Zu jedem $t \in T_N$ existiert nach Vorraussetzung mindestens ein Dienstprozess π_t mit $\pi \models \xi^{\max}_{Org}$, was $\pi \models \hat{\psi}(t)$ impliziert, da $\xi^{\max}_{Org} \Longrightarrow \hat{\psi}(t)$ gilt.

Wir zeigen, dass dieser Dienstprozess π_t die Wohlgeformtheitsbedingung erfüllt: Sei $(K,\phi) \in \mathcal{K}^{pg}(N,\{p\})$ ein Prozess zu beliebigen $R \in \mathcal{R}$ und $p \in P_{Org}(R)$, dann gilt für alle $e \in E_K$:

$$\begin{array}{rclcl} \hat{\psi}_K(e) & = & \bigwedge_{(e',e)\in F^*_K} \psi_K(e') & = & \bigwedge_{(e',e)\in F^*_K} (\psi\circ\phi_{|E})(e') \\ & = & \bigwedge_{(e',e)\in F^*_K} \psi(\phi_{|E}(e')) & = & \bigwedge_{(e',e)\in F^*_K,\phi(e')=t'} \psi(t') \quad = \quad \bigwedge_{t'\in T_e} \psi(t') \end{array}$$

Hierbei definieren wir $T_e := \phi^{-1}(\_ F^*_K e) = \{t \in T \mid \exists e' \in E_K : (e',e) \in F^*_K, \phi(e') = t\}$. Aus $T_e \subseteq T$ folgt:

$$\xi^{\max}_{Org} = \bigwedge_{t\in T_N} \psi(t) \Longrightarrow \bigwedge_{t\in T_e} \psi(t) = \hat{\psi}_K(e)$$

Also gilt auch $\pi_{\phi(e)} \models \hat{\psi}_K(e)$ für alle Ereignisse $e \in E_K$ aller Prozesse (K,ϕ), d.h. die Wohlgeformtheitsbedingung ist erfüllt. q.e.d.

Sei *Org* eine wohlgeformte Organisation mit dem Organisationsnetz N. Zu jeder Transition t gibt es verschiedene Markierungen und Prozesse, in denen t vorkommt. Die Menge aller globalen Steuerungen ist dann:

$$\Xi_{Org}(t) := \{\hat{\psi}_K(e) \mid p \in P, (K,\phi) \in \mathcal{K}^{pg}(N,\{p\}), e \in E_K, \phi(e) = t\}$$

Damit ist Wohlgeformtheit konstruktiv entscheidbar, denn es gilt folgende Aussage.

Theorem 38 *Sei $Org = (N,\mathcal{O},R,D,\psi)$ eine wohlgeformte, koordinierende Organisation.*

1. *Genau dann wenn zu jedem $t \in T_N$ und zu jedem $\xi \in \Xi_{Org}(t)$ ein Prozess $\pi \in Proc(D(t), \xi)$ existiert, dann ist Org wohlgeformt.*

2. *Für alle $t \in T$ eines Organisationsnetzes ist $\Xi_{Org}(t)$ ist eine endliche, konstruktive Menge.*

Beweis: Die erste Eigenschaft folgt direkt aus Definition 49. Dass $\Xi_{Org}(t)$ eine endliche Menge ist, folgt aus der Tatsache, dass $\mathcal{K}^{pg}(Org)$ dies ist. q.e.d.

Kann nicht sichgestellt werden, dass jeder Prozess korrekt ist, so sind entweder einige – oder alle – Regeln $\psi(t)$ zu verschärfen oder es muss verhindert werden, dass dieses Team gewählt werden kann (siehe auch den Abschnitt 5.2 zu Transformation von Organisationen). Eine koordinierende Organisation darf nur korrekte Prozesse zulassen. Natürlich kann man Wohlgeformtheit leicht erreichen, wenn man bereits alle proaktiven Transitionen auf korrekte Prozesse einschränkt. Dabei wird jedoch im allgemeinen das mögliche Verhalten der nachfolgenden Positionen sehr stark eingeschränkt, was deren Flexibilität einschränkt. Hier ist also ein Ausgleich zwischen Wohlgeformtheit und Flexibilität zu erzielen.

4.2.2 Stabilität der Koordinierung

In der Definition der Wohlgeformtheit (Definition 44) wird für ein Multiagentensystem gefordert, dass zu jede Gruppe G einen gemeinsamen Teamplan π_G des Teamnetzes $D(G)$ auswählt und dass dieser Teamplan die Steuerbedingung $\hat{\psi}(G)$ des Teams erfüllt. Dieser Teamplan π_G muss zudem noch einen korrekt terminierten Ablauf des Dienstes beschreiben. Letzteres ist dann besonders einfach zu sicherzustellen, wenn das durch $\hat{\psi}(G)$ gesteuerte Teamnetzes $D(G)$ schwach korrekt ist (vgl. Definition 30).

Wir sind nun daran interessiert, ob die Organisation nur korrekte gesteuerte Gruppen generiert. Dabei müssen wir berücksichtigen, dass jedes Team G von einem Prozess (K, ϕ) des Organisationsnetzes abgeleitet wird, d.h. $G = G(K, \phi)$ für einen Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ mit $p \in P_{Org}(R)$ und $R \in \mathcal{R}$. Organisationen mit dieser Eigenschaft nennen wir *koordinationsstabil*.

Theorem 39 *Sei $Org = (N, \mathcal{O}, R, D, \psi)$ eine koordinierende Organisation. Es ist entscheidbar, ob Org koordinationsstabil ist, d.h. ob für alle $R \in \mathcal{R}$, $p \in P_{Org}(R)$ und $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ gilt, dass $D(G(K, \phi)) \times \hat{\psi}(G(K, \phi))$ ein schwach korrektes Dienstnetz ist.*

Beweis: Da die Menge aller Delegationsprozesse $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ einer wohlgeformten Organisation endlich sind, kann die Eigenschaft der Koordinationsstabilität überprüft werden. q.e.d.

4.3 Organisationspositionen als Agenten

Im folgenden entwicklen wir ein dualistisches Agentenmodell, indem wir auch die Positionen durch Agenten realisieren. Diese Agenten integrieren Handlungsablauf- und Organisationsstrukturbeschreibungen. Wir schließen damit an die Ergebnisse aus Kapitel **??** an, in dem wir gezeigt haben, dass Makro- und Mikroperspektive in einem dualistischen Verhältnis zueinander stehen und nur zwei Seiten der selben

Medaille sind. Auch aus dieser Betrachtungsweise ist folgerichtig, dass die strukturellen Elemente, wie die Positionen, und die interaktiven Elemente, wie die Akteure, durch das gleiche Konstrukt beschrieben werden.

Organisationale Koordination- und individuelle Handlungspläne stehen in wechselseitiger Beziehung, in dem die organisationalen Koordinierungspläne die Handlungspläne der Organisationsmitglieder rahmen und umgekehrt die Handlungen der Mitglieder die Organisationsabläufe mit Leben erfüllen.[2]

Wir betrachten nun Positionen als Agenten, indem wir uns auf die lokale Perspektive, die eine Position O bezüglich der Organisation besitzt, beschränken. In dieser Perspektive sind nur die Position sichtbar, zu denen O eine Verbindung besitzt. Außerdem blenden wir, wie in Abbildung 4.8 gezeigt, den internen Aufbau der umgebenden Positionen aus, da dieser für O nicht sichtbar ist.

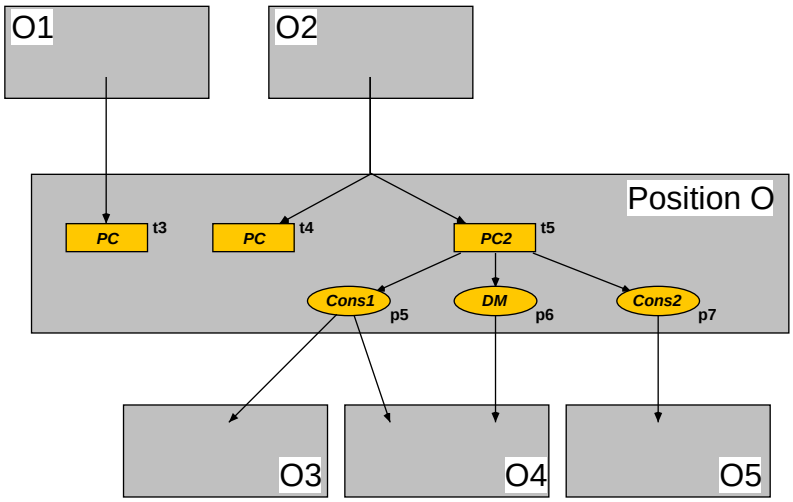


Abbildung 4.8: Lokale Perspektive einer Position

Mit jeder Position ist eine Menge von Rollen verbunden. Als Rollenprofil einer Position $O \in \mathcal{O}$ bezeichnen wir die Menge der Rollen, die die Position anderen gegenüber bereitstellt:

$$R(O) := R({}^{\bullet}(O \cap T) \setminus O) \tag{4.4}$$

Der Organisationseinheit $O \in \mathcal{O}$ wird ein Agent $Ag(O)$ zugewiesen. Diese Agenten nennen wir *Positionsagenten*. Der der Position O zugeordnete Positionsagent ist ein

[2] An dieser Stelle erneut eine Bemerkung zur ontologischen Stellung von Organisationen: Die Aktvität der Organisation ergibt sich stets im konkreten Zusammenspiel von Formalstruktur und Mitgliedern. Und obwohl eine Organisation ohne Mitglieder keine Handlungen hervorbringen kann, wäre es demnach falsch, ihr den Akteursstatus abzusprechen. Dafür lassen sich zwei Argumente ins Feld führen. Das erste Argument behandelt den Kommunikationsaspekt: Andere Akteure können sich in ihrem Handeln auf die Organisation beziehen, ohne dazu Mitglied sein zu müssen. Die Möglichkeit einer Kommunikationsbeziehung zeigt hier die soziale Existenz. In diesem Fall zeigt sich der Akteursstatus durch die Mitgliedschaft in den durch den Beobachter rekonstruierten Kommunikationsprozessen. Das zweite Argument besagt, dass Organisationen Akteure sind, weil ihre Verfahrensregelungen eine eigene Dynamik besitzen. Auch wenn diese Dynamik von den Mitgliedern erzeugt wird, ist sie nicht den Mitgliedern zuzurechnen, sondern der Entität Organisation, da diese der primärer Ursprung der Handlungslogik ist. Auch hier zeigt sich die Organisation dem Beobachter als eigenständiger Akteur, mit dem die Organisationsmitglieder interagieren.

Agent, der das Rollenprofil $R(O)$ einnimmt.[3] Wie wir in Kapitel 2 bereits festgestellt haben, spezifizieren Rollenkomponenten von Dienstnetzen die Rechte und Pflichten einer Rolle. Ebenso definieren Positionen ebenfalls Rechte und Pflichten. Die Rechte einer Position definieren sich über die Delegationsmöglichkeiten, Pflichten sind über die Rollen $R(O)$ definiert.

Jede Position muss mit einem Agenten besetzt werden, der das Rollenprofil wahrnehmen kann. Um das Rollenprofil wahrnehmen zu können, stehen der Position die ihr zugewiesenen Dienste zur Vefügung:

$$D(O \cap T) = \{D(t) \mid t \in (O \cap T)\}$$

Man beachte, dass der Dienst, der bei der Auswahl der Tätigkeit t ausführt wird, nur in der Einschränkung $D(t)[R(t^\bullet)]$ genutzt wird, d.h. nur der Anteil der Interaktion, die die genutzten Rollenprofile betrifft.

Beispiel Betrachten wir die Organisation aus der Abbildung 4.1. Das Rollenprofil $R(O)$ ist in der Abbildung an den Positionen annotiert. Die Positionen definieren folgende Rollenprofile: $R(O_1) = \{Prod, Cons\}$, $R(O_2) = \{Prod\}$, $R(O_3) = \{Cons\}$ und $R(O_4) = \{Cons, Cons_1\}$. Sie implementieren folgende Dienste: $D(O_1) = \{PC\}$, $D(O_2) = \{PC_3\}$ und $R(O_3) = R(O_4) = \{PC, PC_2\}$.

Die Tätigkeit, die t_5 realisieren muss, ist nur

$$D(t_5)[t_5{}^\bullet] = PC_2[\{Cons_1, Cons_2, DM\}],$$

umfasst also nicht den Konsumentenanteil, da nur das Rollenprofil $R(p_2) = \{Cons\}$ der Position p_2 im Vorbereich von t_5 relevant ist. ◇

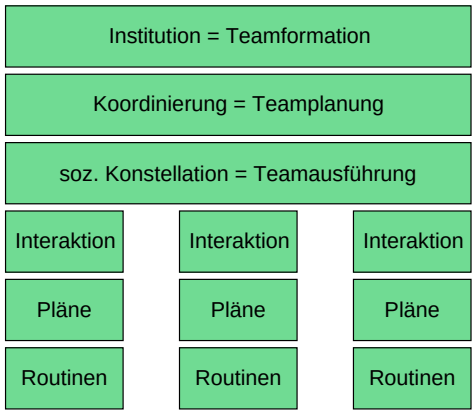


Abbildung 4.9: Positionen als Agenten

Die Art und Weise, wie Positionen agieren, entspricht denen der SONAR-Agenten (vgl. dazu Abbildung 4.9). Die organisationalen Routinen, die den Teamformationsprozess steuern, sind durch das Organisationsnetz festgelegt. Jedes Teamnetz G koordiniert die Teamplanung, welche das organisationale Pendant zur Akteursplanung darstellt. Die Ausführung des gemeinsamen Teamplans π_G als spezifische Form der sozialen Konstellation korrespondiert zur Interaktivität des Akteurs. Wir erkennen also in Abbildung 4.9 die in Abbildung **??** dargestellte selbstähnliche Korrespondenz von Mikro- und Makroprozessen wieder.

[3] Von $R(O)$ ist die Menge $R(O \cap P) = \{R(p) \mid p \in (O \cap P)\}$ zu unterscheiden, d.h. die Menge der Rollen, die eine Position zur Ausführung der mit seinem Profil verbundenen Dienste nutzt.

4.4 Multiagentensystemorganisationen

Wir betrachten die Ausführungsumgebung, mit deren Hilfe die Agenten und ihre Pläne in organisationale Abläufe und Strukturen eingebettet werden. Betrachten wir eine Organisation $Org = (N, \mathcal{O}, R, D, \psi)$. Eine *MAS-Organisation* ist eine Instanz einer formalen Organisation. Jeder Organisationseinheit $O \in \mathcal{O}$ wird dazu der Agentenname A_O und der Positionsagent $Ag(O)$ zugewiesen. Jede MAS-Organisation generiert ein Multiagentensystem, das generierte SONAR-*MAS*, indem man jeder Organisationsposition $O \in \mathcal{O}$ den Agenten A_O zuweist.

Für eine gegebene Organisation *Org* setze die Menge an Agentennamen zu $\mathcal{A} = \{A_O \mid O \in \mathcal{O}\}$, die Dienstklasse zu $\mathcal{D} = \{D(t) \mid t \in T_{Org}\}$, die Rollenstruktur zu $\mathcal{R} = \{R(D) \mid D \in \mathcal{D}\}$ und die Menge an Gruppennetzen zu $\mathcal{G} = \mathcal{G}(Org)$. Damit ist die Menge der SONAR-Agenten $\mathcal{S}_{(\mathcal{A},\mathcal{R},\mathcal{D},\mathcal{G})}$ festgelegt.

Definition 50 *Eine* MAS-Organisation (Org, Ag) *besteht aus*

1. *einer koordinierten Organisation* $Org = (N, \mathcal{O}, R, D, \psi)$ *und*
2. *einer Injektion* $Ag : \mathcal{O} \to \mathcal{S}_{(\mathcal{A},\mathcal{R},\mathcal{D},\mathcal{G})}$*, die jeder Position einen Positionsagenten zuweist.*

Sei (Org, Ag) *eine MAS-Organisation, dann bezeichnet*

$$\mu(Org, Ag) := (\{A_O \mid O \in \mathcal{O}\}, Ag)$$

das von ihm generierte SONAR-MAS.

Für jede Stelle $p \in P$ *des Organisationsnetzes bezeichnet* $\mathcal{O}_{dlg}(p)$ *die Menge aller Positionen und* $\mathcal{A}_{dlg}(p)$ *die Menge aller Agenten, an die die Bearbeitung delegiert werden kann:*

$$\begin{array}{lcll} \mathcal{O}_{dlg}(p) & := & O(p^\bullet) & = & \{O(t) \mid t \in p^\bullet\} \\ \mathcal{A}_{dlg}(p) & := & Ag(\mathcal{O}_{dlg}(p)) & = & \{Ag(O(t)) \mid t \in p^\bullet\} \end{array} \tag{4.5}$$

Für jede Transition $t \in T$ *des Organisationsnetzes bezeichnet* $\mathcal{O}_{gen}(t)$ *die Menge aller Positionen und* $\mathcal{A}_{gen}(t)$ *die Menge aller Agenten, die Aufträge für* t *generieren können:*

$$\begin{array}{lcll} \mathcal{O}_{gen}(t) & := & O(^\bullet t) & = & \{O(p) \mid p \in {}^\bullet t\} \\ \mathcal{A}_{gen}(t) & := & Ag(\mathcal{O}_{gen}(t)) & = & \{Ag(O(p)) \mid p \in {}^\bullet t\} \end{array} \tag{4.6}$$

Wir weisen darauf hin, dass die Organisationsstruktur sich sehr unterschiedlich auf die Interaktionsprozesse auswirken kann. So ist es möglich, dass zwei Positionsagenten miteinander in Teamplanungsprozessen miteinander interagieren, obwohl sie nach dem Organisationsnetz in keiner Verbindung miteinander stehen. Dies liegt daran, dass ein Organisationsnetz die Positionen bezüglich der Delegationsmöglichkeiten relationiert, nicht aber bezüglich der Interaktivität. Ein Beispiel stellt die Organisationsform des Marktes ist Abbildung 4.3 dar: Nach dem Organisationsnetz stehen Erzeuger und Verbraucher in keiner Delegationsbeziehung. Gleichwohl ist jeweils ein Erzeuger und ein Verbraucher Mitglied in den generierbaren Teams, in denen sie auch miteinander interagieren.

Ist eine MAS-Organisation (Org, Ag) gegeben, so kann anhand eines gegebenen Prozesses (K, ϕ) ein Gruppennetz nach Definition 40 konstruiert werden. Dazu ist die Agentenabbildung A_G angeben. Jede Aktivität e im Prozess wird auf die Tätigkeit $\phi(e)$ abgebildet, die sich in der Position $O(\phi(e))$ befindet, die vom Positionsagenten $Ag(O(\phi(e)))$ implementiert wird. Dies ist der Agent $A_G(e)$, der e zuzuweisen ist.

Theorem 40 *Sei $Org = (N, \mathcal{O}, R, D, \psi)$ eine koordinierende Organisation und $R \in \mathcal{R}$ eine Rolle. Dann generiert jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ zu einem $p \in P_{Org}(R)$ die Gruppe:*

$$G(K, \phi) := (K, (R \circ \phi), (D \circ \phi), \psi \circ \phi_{|E}, (Ag \circ O \circ \phi))$$

Beweis: Wir wissen bereits aus Theorem 34, dass $(K, (R \circ \phi), (D \circ \phi), (\psi \circ \phi_{|E}))$ ein Teamnetz ist.

Nach Definition 40 muss für die Besetzung A eines Gruppennetzes $A(p) = A(t)$ für alle $p \in t^\bullet$ gelten. Wie bereits dargelegt ist zudem durch $A = (Ag \circ O \circ \phi)$ eine Besetzung definiert, die zur Organisationsstruktur passt, denn in einer Position O gilt nach Definition $t \in O \Longrightarrow t^\bullet \subseteq O$. q.e.d.

4.5 Konstruktion wohlorganisierter Multiagentensysteme

Im folgenden zeigen wir, dass eine Organisation genug Struktur enthält, um ein Multiagentensystem generativ zu definieren, das genau der formalen Organisation entspricht. Wir betrachten konkret, wie man anhand einer gegebenen Organisation ein wohlgeformtes, wohlkoordinierendes SONAR-MAS konstruieren kann.

Zwischen der Organisation und den Positionsagenten müssen Konsistenzbeziehungen erhalten sein. Dies umfasst somit die Tätigkeiten, die Profile, die Rollen und das Organisationsnetzwerk. Die Rollenfähigkeiten ρ_A und die Handlungsmöglichkeiten sind durch die Organisation implizit festgelegt. Jeder Dienst D kann vom Agenten A durch solche Tätigkeiten geleistet werden, die erstens dem Agenten A zugeordnet sind (d.h. $t \in Ag^{-1}(A) \cap T$ muss gelten) und die zweitens in der Organisation D zugeordnet sind (d.h. $D(t) = D$ muss gelten). Für jede solche Tätigkeiten t muss A dann mindestens die Rollen der Positionen im Nachbereich $t^\bullet$ realisieren.

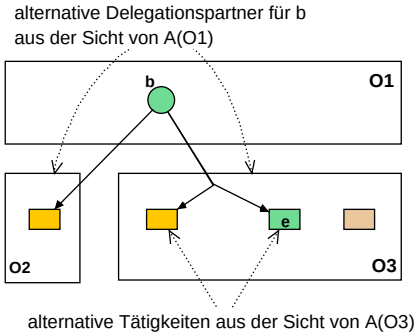


Abbildung 4.10: Auswahlpunkte

Jeder Agenten kodiert in $\Delta_A(G)(b)$ Handlungsmöglichkeiten, um auf die Rolle zu reagieren. Ist das Team G ein Prozess von N, so ist b die Stelle $p = \phi(b)$ zugewiesen. Als Tätigkeiten kommen in der Organisation jetzt solche in Frage, die erstens dem Agenten A zugeordnet sind (d.h. $t \in Ag^{-1}(A) \cap T$ muss gelten) und die zweitens im Nachbereich von $p = \phi(b)$ liegen (d.h. $t \in \phi(b)^\bullet$ muss gelten). Dies ist in Abbildung 4.10 illustriert. Mit der Wahl einer Tätigkeit t sind durch den Nachbereich bereits die Rollen der Partner festgelegt: Für jedes Rollenprofil $p \in t^\bullet$ ist die Rolle $R(p)$ zu besetzen. Wer diese Rolle implementiert, ist durch die Organisation ebenfalls eingeschränkt, denn es kommen Agenten in Frage, für die

eine Tätigkeit $t \in p^\bullet$ existiert und die in der Organisation vom Agenten $Ag(O(t)))$ ausgeführt werden, d.h. Agenten aus der Menge $\mathcal{A}_{dlg}(p)$.

Diese zuvor formulierten Bedingungen integrieren wir nun im folgenden in das formale Modell.

4.5.1 Prozessentfaltung

Teamplanung stellt eine Prozessentfaltung des Organisationsnetzes dar. Die Prozesse eines Netzes $N = (P, T, F, m_0)$ lassen sich induktiv charakterisieren:

1. Sei C eine Menge mit $\phi(C) = M_0$, dann ist $((C, \emptyset, \emptyset), \phi)$ ein Prozess von N.

2. Sei $(K, \phi) = ((B, E, F), \phi)$ ein Prozess, der für ein $t \in T$ eine Menge $B_t \subseteq K^\circ$ enthält, die auf die Eingangsstelle ${}^\bullet t$ abgebildet wird, d.h. es gilt $\phi(B_t) = {}^\bullet t$.

   Dann bilden wir die Prozessverlängerung $((B', E', F'), \phi')$ indem wir ein frisches Ereignis e mit $\phi(e) = t$ an alle Stellen in B_t anhängen und eine frische Menge von Bedingungen $B'_t = \{b_p \mid p \in t^\bullet\}$ mit $\phi(b_p) = p$ an das Ereignis e anhängen.

Für Prozesse unter der Fortschrittseigenschaft wird dieses Verfahren solange angewendet, bis es keine Menge $B_t \subseteq K^\circ$ mehr gibt, die auf die Eingangsstelle ${}^\bullet t$ abgebildet wird. Abbildung 4.11 zeigt den Algorithmus zur Konstruktion aller endlichen Prozesse unter der Fortschrittseigenschaft. Der Algorithmus ist nichtdeterministisch in der Auswahl der Transition t und der Menge B_t. Das Verfahren terminiert, wenn der Prozess nicht weiter verlängerbar ist.

---

function $K(N, p)$ is
  $(K, \phi) := ((\{b_0\}, \emptyset, \emptyset), \{(b_0, p)\})$
  while $\exists t \in T : \exists B_t \subseteq K^\circ : \phi(B_t) = {}^\bullet t$ do
    $(K, \phi) := ((B', E', F'), \phi')$ where
      $B'_t = \{b_p \mid p \in t^\bullet\}$
      $B' = B_K \uplus B'_t$
      $E' = E_K \uplus \{e\}$
      $F' = F_K \cup (B_t \times \{e\}) \cup (\{e\} \times B'_t)$
      $\phi' = \phi \cup \{(e, t)\} \cup \{(b_p, p) \mid p \in t^\bullet\}$
  endwhile
  return (K, ϕ)
end

Abbildung 4.11: Algorithmus zur induktiven Prozesserzeugung unter der Fortschrittseigenschaft

4.5.2 Organisationssicht der Teamformation

Teamplanung ähnelt der induktiven Prozesserzeugung. Sei das Organisationsnetz N mit der Rolle R gegeben. Sei p eine Stelle aus $P_{Org}(R)$, die das Rollenprofil R besitzt. In der Organisation beginnt der Agent $A_p = Ag(O(p))$ mit der Bearbeitung. Initial definiert A_p das Anfangsstück K, das nur aus einer Stelle b_0 besteht, für die $R_G(b_0) = R_0$ gilt und die durch ϕ auf eine initiale Stelle des Organisationsnetzes

abgebildet wird, d.h. es gilt $A_p = Ag(O(p)) = Ag(O(\phi(b_0)))$. Dies ist der initiale Teampräfix.

Zu einem Anfangsstück eines Teams K wird das Team an den Aufgaben $K^\circ \cap B$ weiter entwickelt. Zu jedem $b \in K^\circ \cap B$ ist ein Agent A^b festzulegen, an den die Aufgabe delegiert wird. Dieser Agent A^b wird vom Agenten $A_b = Ag(O(\phi(b)))$ festgelegt.

Da die Teambildungsprozesse miteinander konsistent sein sollen, betrachten wir nun, wie A^b den Präfix weiter entwickelt, denn die Wahl von $Ag(O(\phi(b)))$ muss hiermit identisch sein.

An die Stelle b wird von A^b ein neues Ereignis e angefügt, wobei – aus Organisationssicht – für das Bild $\phi(e)$ nur Transitionen in Frage kommen, die im Nachbereich von $\phi(b)$ liegen und dem gleichen Agenten A^b zugeordnet sind:

$$\phi(e) \in \left(\phi(b)^\bullet \cap Ag^{-1}(A^b)\right)$$

An das Ereignis e werden die Bedingungen $B'_t = \{b_p \mid p \in t^\bullet\}$ gehängt und es wird $\phi(b_p) = p$ gesetzt.

---

function $K(N,p)$ is
  $(K,\phi) := ((\{b_0\},\emptyset,\emptyset),\{(b_0,p)\})$
  while $(K^\circ \cap B) \neq \emptyset$ do
    for each $b \in (K^\circ \cap B)$ do
      $(K,\phi) := ((B',E',F'),\phi')$ where
        $A^b \in \mathcal{A}_{dlg}(\phi(b))$
        $t \in (\phi(b)^\bullet \cap Ag^{-1}(A^b))$
        $B'_t = \{b_p \mid p \in t^\bullet\}$
        $B' = B_K \uplus B'_t$
        $E' = E_K \uplus \{e\}$
        $F' = F_K \cup \{(b,e)\} \cup (\{e\} \times B'_t)$
        $\phi' = \phi \cup \{(e,t)\} \cup \{(b_p,p) \mid p \in t^\bullet\}$
  endwhile
  return (K,ϕ)
end

Abbildung 4.12: Nichtdeterministischer Teambildungsalgorithmus

Abbildung 4.12 zeigt den Algorithmus zur Teambildung. Der Algorithmus ist nichtdeterministisch in der Auswahl der Transition t und des Agenten A^b. Das Verfahren terminiert, wenn für jede maximale Stelle eine Transition t mit $t^\bullet = \emptyset$ gewählt wird, da dann $(K^\circ \cap B)$ die leere Menge ist. Der konstruierte Prozess beschreibt einen Tätigkeitspfad (vgl. Theorem 27). Als Ergebnis der Teambildung erhalten wir ein Kausalnetz (K,ϕ), das das Team $G(k,\phi)$ repräsentiert.

Theorem 41 *Sei $Org = (N,\mathcal{O},R,D)$ eine wohlgeformte Organisation.*

1. *Ist die Markierung $\{p\}$ in N sicher bearbeitbar, dann hat Algorithmus 4.12 die Option zu terminieren.*

2. *Ist N sicher bearbeitbar, dann hat der Algorithmus 4.12 für alle p die Option zu terminieren.*

3. *Ist die Markierung $\{p\}$ in N sicher bearbeitbar, dann ist $\mathcal{K}^{pg}(N, \{p\})$ eine nicht-leere Menge.*

Beweis: Wir beweisen die einzelnen Teile in der angegebenen Reihenfolge.

1. Wenn die Markierung $\{p\}$ in N bearbeitbar ist, dann gilt $\mathbf{0} \in RS(N, m)$ für alle m, die von $\{p\}$ erreichbar sind. Also gibt es für alle m Schaltfolgen, so dass $\{p\} \xrightarrow{w_1} m \xrightarrow{w_2} \mathbf{0}$. Da der Algorithmus 4.12 genau dann terminiert, wenn $\mathbf{0}$ erreicht wird und er für jede erreichbare Markierung m die Möglichkeit besitzt diese zu erreichen, besteht auch stets die Option zu terminieren.

2. Die Aussage folgt aus (1).

3. Nach Proposition 24 ist die Markierung $\{p\}$ genau dann bearbeitbar, wenn das Nonterminal A_p der kontextfreien Grammatik $G(N, m)$ produktiv ist und nach Proposition 27 ist $\mathcal{K}^{pg}(N, \{p\})$ genau dann eine nicht-leere Menge, wenn A_p produktiv in $G(N, \{p\})$ ist.

q.e.d.

Der Teambildungsalgorithmus 4.12 generiert alle möglichen Prozesse.

Theorem 42 *Sei $Org = (N, \mathcal{O}, R, D)$ eine wohlgeformte Organisation.Für alle $R \in \mathcal{R}$ und $p \in P_{Org}(R)$ gilt, dass jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, p)$ vom Teambildungsalgorithmus in Abbildung 4.12 generiert werden kann.*

Beweis: Jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, p)$ kann auch induktiv vom Algorithmus 4.11 zur induktiven Prozesserzeugung erzeugt werden. Beachtet man, dass für R/D-Netze $|{}^\bullet t| = 1$ und daher auch $|B_t| = 1$ gilt, so stellt man fest, dass sich dieser Algorithmus kaum vom Teambildungsalgorithmus 4.12 unterscheidet.

Der einzige Unterschied besteht darin dass die Transition t nicht aus der Menge $\phi(b)^\bullet$, sondern nur aus $\phi(b)^\bullet \cap Ag^{-1}(A^b)$ gewählt wird.

Da die Wahl $A^b \in \mathcal{A}_{dlg}(\phi(b))$ beliebig ist, können wir die Vereinigung über alle möglichen A^b betrachten:

$$\bigcup_{A^b \in \mathcal{A}_{dlg}(\phi(b))} \left(\phi(b)^\bullet \cap Ag^{-1}(A^b)\right) = \phi(b)^\bullet \cap \left(\bigcup_{A^b \in \mathcal{A}_{dlg}(\phi(b))} Ag^{-1}(A^b)\right) = \phi(b)^\bullet$$

Die zweite Umformung gilt, da mit $\mathcal{A}_{dlg}(p) = \{Ag(t) \mid t \in p^\bullet\}$ bereits $\phi(b)^\bullet \subseteq \bigcup_{A^b \in \mathcal{A}_{dlg}(\phi(b))} Ag^{-1}(A^b)$ gilt.

Also hat der Teambildungsalgorithmus die gleiche Auswahl wie der Algorithmus zur induktiven Prozesserzeugung. Dann stimmen also auch die generierten Prozesse überein. q.e.d.

Den Nichtdeterminismus in Algorithmus 4.12 können wir eleminieren, indem wir das Abbildungstupel $\tau = (\tau^1, \tau^2, \tau^3)$ definieren, mit deren Hilfe der Agent $A_b = (Ag \circ O \circ \phi)(b)$ die Auswahlen $A^b \in \mathcal{A}_{dlg}(\phi(b))$ und $t \in \phi(b)^\bullet \cap Ag^{-1}(A^b)$ trifft. Da wir die Prozesse eigentlich nicht anhand einer Markierung $m = \{p\}$ generieren, sondern zu einer zu implementierenden Rolle R, treffen wir außerdem noch eine Auswahl $p \in P_{Org}(R)$:

$$\begin{aligned} A^b &= \tau^1(K, \phi, b) \in \mathcal{A}_{dlg}(\phi(b)) \\ t &= \tau^2(K, \phi, b) \in (\phi(b)^\bullet \cap Ag^{-1}(A^b)) \\ p &= \tau^3(R) \in P_{Org}(R) \end{aligned}$$

Auch wenn die Wahl $\tau(K, \phi, b)$ bei der Bearbeitung von b eigentlich nur vom zuständigen Agenten $A_b = (Ag \circ O \circ \phi)(b)$ getroffen wird, muss die Auswahl in allen beteiligten Agenten identisch sein, damit die Formation der Subteams miteinander konsistent verläuft.

Definition 51 *Das Abbildungstupel $\tau = (\tau^1, \tau^2, \tau^3)$ heißt* Teamkonstruktor*, wenn die Formation der Subteams konsistent verläuft:*

$$\forall A \in \mathcal{A}_\mu : \tau_A(K, \phi, b) = \tau_{(Ag \circ O \circ \phi)(b)}(K, \phi, b) \tag{4.7}$$

Bei festgelegten Teamkonstruktor τ ist – im Falle der Termination – der vom Algorithmus 4.13 generierte Prozess eindeutig festgelegt. Diesen Prozess bezeichnen wir als $K_\tau(N, p)$. Ist eine Rolle R vorgegeben, so definieren wir:

$$K_\tau(N, R) := K_\tau(N, \tau^3(R)) \tag{4.8}$$

Für die Eindeutigkeit der Konstruktion ist zentral, dass die Reihenfolge der Auswahl der $b \in (K^\circ \cap B)$ keine Auswirkung auf das Ergebnis hat, da der Prozess für jedes b an disjunkten Stellen weiter entwickelt wird.

---

function $K_\tau(N, p)$ is
  $(K, \phi) := ((\{b_0\}, \emptyset, \emptyset), \{(b_0, p)\})$
  while $(K^\circ \cap B) \neq \emptyset$ do
    foreach $b \in (K^\circ \cap B)$ do
      $(K, \phi) := ((B', E', F'), \phi')$ where
        $A_b = (Ag \circ O \circ \phi)(b)$
        $A^b = \tau^1_{A_b}(K, \phi, b)$
        $t = \tau^2_{A^b}(K, \phi, b)$
        $B'_t = \{b_p \mid p \in t^\bullet\}$
        $B' = B_K \uplus B'_t$
        $E' = E_K \uplus \{e\}$
        $F' = F_K \cup \{(b, e)\} \cup (\{e\} \times B'_t)$
        $\phi' = \phi \cup \{(e, t)\} \cup \{(b_p, p) \mid p \in t^\bullet\}$
    end
  endwhile
  return (K, ϕ)
end

Abbildung 4.13: Algorithmus zur Teambildung $K_\tau(N, p)$

Die Aussage von Theorem 42 gilt auch für die deterministische Konstruktion mit Hilfe des Konstruktors τ nach Algorithmus 4.13.

Theorem 43 *Sei der Teamkonstruktor τ gegeben. Durch Variation von τ können alle Prozesse erzeugt werden:*

$$\bigcup_\tau K_\tau(N, p) = \mathcal{K}^{pg}(N, p)$$

Beweis: Dazu müssen wir nur beachten, dass wir durch Variation von τ alle Auswahlen realisieren können. Wir erhalten somit die folgende Aussagen:

$$\bigcup_\tau \tau^1(K, \phi, b) = \mathcal{A}_{dlg}(\phi(b))$$

und mit $A_b = (Ag \circ O \circ \phi)(b)$ und $A^b = \tau^1_{A_b}(K, \phi, b)$:

$$\bigcup_{\tau} \tau^2(K, \phi, b) = \phi(b)^\bullet \cap Ag^{-1}(A^b)$$

Also können alle Prozesse erzeugt werden. q.e.d.

Die Variation des Teamkonstruktors τ ist bereits als Lernaktivität im SONAR-Modell vorgesehen, nämlich als die Meta-Planung der Inkorporation. Hierbei wird die Teamgenerationsfunktion γ modifiziert. Da γ aber mit Hilfe des Teamkonstruktors τ definiert wird, verändert die Meta-Planung auch den Teamkonstruktor.

4.5.3 Agentensicht der Teamformation

Aus der Sicht des Agent A^b, dem die Aufgabe b zur Bearbeitung zugewiesen wurde, wird die Auswahl nicht unter den Elementen, die der Teambildungsalgorithmus betrachtet:

$$\phi(b)^\bullet \cap Ag^{-1}(A^b),$$

sondern unter den Elementen seiner Präferenz:

$$\Delta_{A^b}(b)$$

getroffen. Um ein an die Organisation angepasstes Multiagentensystem zu erhalten, konstruieren wird Δ_A und δ_A derart, dass diese beiden Sichtweisen gerade zusammenfallen.

Sei $Org = (N, \mathcal{O}, R, D, \psi)$ eine wohlgeformte Organisation. In einer MAS-Organisation (Org, Ag) sind die notwendigen Rollenfähigkeiten ρ_A, die Teamgeneration γ_A und die Handlungsmöglichkeiten Δ_A, δ_A durch die Organisation implizit festgelegt.

Definition 52 *Eine MAS-Organisation (Org, Ag) mit $Org = (N, \mathcal{O}, R, D, \psi)$ heißt* wohlgeformt*, wenn für jeden Positionsgagenten A_O mit $O \in \mathcal{O}$ ein Teamkonstruktor τ_{A_O} existiert, für den folgendes gilt:*

1. *Die benötigten Rollenfähigkeiten ρ_{A_O} ergeben sich anhand der A_O zugeordneten Transitionen:*

$$\rho_{A_O}(D_0) = \{R(t^\bullet) \mid t \in Ag^{-1}(A_O) \cap T, D(t) = D_0\}$$

2. *Die Teamgeneration $\gamma_{A_O} : \mathcal{R} \to \mathcal{G}$ geschieht anhand des Teamplanungsalgorithmus.*

$$\gamma_{A_O}(R) = G(K_{\tau_{A_O}}(N, R))$$

3. *Die Verhaltensregelmäßigkeiten $\Delta_{A_O}(G)(b)$ ergeben sich anhand der A_O zugeordneten Transitionen. Definiere für alle $b \in B_G$:*

$$\begin{aligned}\Delta_{A_O}(G)(b) = \{ \quad & (D(t), \{(R(p), A_p) \mid p \in t^\bullet\}) \\ \mid \quad & t \in (T \cap \phi(b)^\bullet \cap Ag^{-1}(A_O)), \forall p \in t^\bullet : A_p \in \mathcal{A}_{dlg}(p)\}\end{aligned}$$

4. *Die Handlungsmöglichkeiten $\delta_{A_O}(G)(\Delta)(b)$ ergeben sich anhand des Teams G. Definiere für alle $b \in B_G$:*

$$\begin{aligned}\delta_{A_O}(G)(\Delta)(b) := (D(\phi(e)), \{ \quad & (R(\phi(b')), A_{O(\phi(e'_b))}) \mid \\ & (b, e) \in F_G, \forall b' \in e^\bullet : (b', e'_b) \in F_G\})\end{aligned}$$

Beachte, dass im Team G zu jedem b der Nachfolger e eindeutig bestimmt ist, ebenso wie zu jedem $b' \in e^\bullet$ das e'_b.

5. *Der Handlungsplan muss ein beliebiger Teamplan sein, d.h. ein Prozess, der die Steuerbedingung erfüllt:*

$$\pi_{A_O}(G)(D)(p) \in Proc(D(G), \hat{\psi}(G))$$

Die Definition einer wohlgeformten koordinierenden Organisation stellt sicher, dass ein solcher Prozess stets existiert.

Durch *Org* sind mit Definition 52 die Abbildungen ρ und Δ und durch τ sind γ und δ konstruktiv festgelegt. Durch $\hat{\psi}$ wird π eingeschränkt, im allgemeinen jedoch nicht festgelegt.

4.5.4 Konstruktion eines Sonar-MAS

Jede wohlgeformte MAS-Organisation (Org, Ag) generiert ein SONAR-MAS $\mu(Org, Ag)$. Es zeigt sich deutlich, dass die Agenten eines generierten SONAR-MAS die *formale* Organisation modellieren, denn sie entsprechen exakt den Anforderungen der Organisationen, mit anderen Worten: Sie sind mit ihr identisch.

Generierte SONAR-MAS sind praktisch relevant, da sie stets wohlgeformte Multiagentensysteme darstellen und alle gebildeten Prozesse Teams darstellen.

Theorem 44 *Das von einer wohlgeformten MAS-Organisation (Org, Ag) generierte* SONAR*-MAS $\mu(Org, Ag)$ ist wohlgeformt.*

Beweis: In $\mu(Org, Ag)$ ist jedem $O \in \mathcal{O}$ der Agent A_O zugeordnet. Nach Definition 44 sind folgende Bedingungen zu erfüllen:

1. A_O kennt nur Teampartner, die durch die Verhaltensregelmäßgkeiten $\Delta_{A_O}(G)(p)$ über die Organisationsstruktur erzeugt werden, wobei nach Konstruktion nur Agenten aus der Menge $\{Ag(O) \mid O \in \mathcal{O}\}$ verwendet.

2. A_O vefügt nach Konstruktion für jede Präferenz $\Delta_{A_O}(G)(p)$ über die benötigten Fähigkeiten.

3. Der Handlungsplan ist nach Konstruktion immer ein Prozess, der die Steuerbedingung erfüllt.

Also gilt Wohlgeformtheit. q.e.d.

Ob das Multiagentensystem das Team G akzeptiert, hängt von der Auswahl der Verhaltensregelmäßgkeiten ab. Im jedem Schritt des Algorithmus wählt $Ag(O(\phi(b)))$ eine Transition t als Verlängerung von b. Dies ist gleichbedeutend mit der Auswahl einer Verhaltensregelmäßgkeit:

$$\delta := \delta_A(G)(\Delta)(p) = (D, \{(R_1, A_1), \ldots, (R_n, A_n)\})$$

Nach Konstruktion (Definition 52) passt Δ_A und δ bereits zu Struktur von G, so dass wir folgende Aussage erhalten:

Theorem 45 *Sei $Org = (N, \mathcal{O}, R, D, \psi)$ eine wohlgeformte Organisation und sei $R \in \mathcal{R}$, $p \in P_{Org}(R)$ und $\{p\}$ bearbeitbar.*

Wenn τ ein Teamkonstruktor ist, dann wird jedes vom Teambildungsalgorithmus generierte Team $G(K_\tau(N, p))$ auch vom generierten SONAR*-MAS akzeptiert.*

Beweis: Nach Theorem 41 hat der Algorithmus 4.12 die Option zu terminieren, so dass mindestens ein Team $G(K_\tau(N,p))$ vom Teambildungsalgorithmus generiert wird.

Es sind zwei Eigenschaften zu zeigen:

1. Die Subteams passen zusammen: Dies gilt aufgrund der Konsistenzforderung (4.7) des Teamkonstruktors.

2. Die Teamzuweisungen entsprechen den individuellen Präfenzen aller Agenten, d.h. für alle $(b,e) \in F_G$ muss gelten:

$$\delta_{A(e)}(G)(\Delta)(b) = (D_G(e), \{(R_G(b'), A_G(e(b'))) \mid b' \in e^\bullet\})$$

   Dies gilt, da nach Konstruktion im generierten SONAR-MAS gilt:

$$\begin{aligned} & \delta_A(G)(\Delta)(b) \\ = \; & (D(\phi(e)), \{(R(\phi(b')), Ag(O(\phi(e'_b)))) \mid (b,e) \in F_G, \forall b' \in e^\bullet : (b', e'_b) \in F_G\}) \\ = \; & (D_G(e), \{(R_G(b'), A_G(e'_b) \mid (b,e) \in F_G, \forall b' \in e^\bullet : (b', e'_b) \in F_G\}) \\ = \; & (D_G(e), \{(R_G(b'), A_G(e(b')) \mid (b,e) \in F_G, b' \in e^\bullet\}) \end{aligned}$$

   Die erste Umformung ergibt sich, weil die Gruppe aus (K,ϕ) konstruiert wird $(G(K,\phi))$, so dass $R_G = R \circ \phi$, $D_G = D \circ \phi$ und $A_G = Ag \circ O \circ \phi$ gilt. Nutzen wir dann aus, dass die Nachbedingung von b' eindeutig bestimmt ist, so erhalten wir die zweite Umformung.

q.e.d.

Das Theorem zeigt, dass für ein wohlgeformtes generiertes SONAR-MAS die Teamauswahl nicht eingeschränkt ist, da – im Gegensatz zu beliebigen Multiagentensystemen – alle Teams akzeptiert werden. Somit sind generierte SONAR-MAS besonders geeignet, als Vorstrukturierung der Teamprozesse eines Multiagentensystems zu dienen.

Wir erhalten aus Theorem 45 als Korrolar die Aussage, dass generierte Multiagentensysteme organisiert sind.

Theorem 46 *Sei (Org, Ag) eine wohlgeformte MAS-Organisation, dann ist das generierten* SONAR*-MAS $\mu(Org, Ag)$ organisiert.*

Beweis: Das generierten SONAR-MAS $(\mathcal{A}_\mu, \mu) = \mu(Org, Ag)$ hat nach Konstruktion in Definition 52 nur Teamnetze, die von der Netzstruktur der organisation *Org* erzeugt wird: $\mathcal{G} = \mathcal{G}(Org)$. Nach Theorem 45 werden außerdem alle Teamnetze vom Multiagentensystem akzeptiert: $\mathcal{G}(Org) = \mathcal{G}_{akz}(\mathcal{A}_\mu, \mu)$. Damit ist $\mu(Org, Ag)$ im Sinne von Definition 48 organisiert. q.e.d.

Zusammenfassung

Organisationen generieren konstruktiv ein SONAR-MAS, das sich sehr gut zur Vorstrukturierung der Teamprozesse eines Multiagentensystems eignet. Diese Eigenschaft folgt aus der Tatsache, dass alle Prozesse der Organisation ein zugelassenes Team generieren. Ebenso ist von Interesse, dass sich aus einer formalen Organisation stets ein wohlgeformtes Multiagentensystem ableiten lässt. Abbildung 4.14 fasst die Beziehungen zwischen Organisation, Team und Multiagentensystem zusammen.

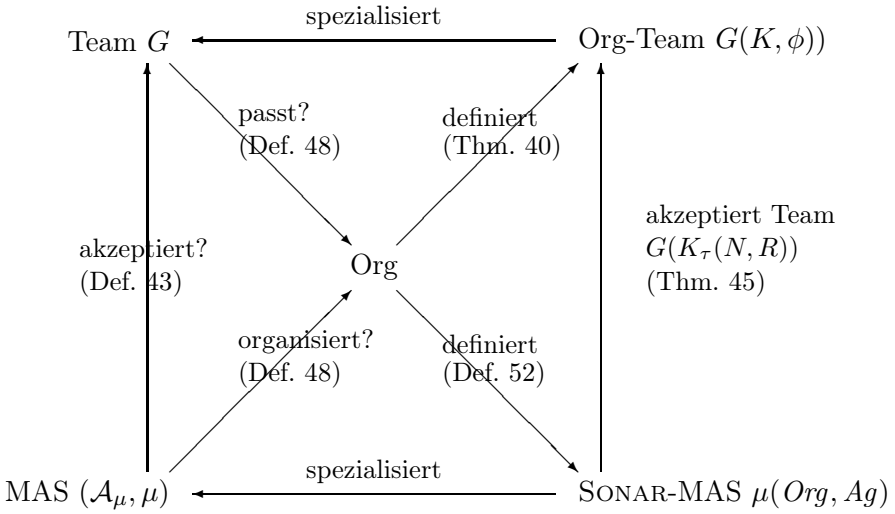


Abbildung 4.14: Beziehungen zwischen Organisation, Team und Multiagentensystem

Organisationale Planung	Modellelement
strategische Ebene	Organisationsstruktur *Org*
taktische Ebene	Teamformation G
operationale Ebene	Teamplanung π_G

Tabelle 4.1: Die Aktivitätsebenen der organisationalen Planung

An dieser Stelle zeigt sich, dass das Modell der formalen Organisation hilft, organisationale Planung auf der strategischen, der taktischen und der operationalen Ebene zu unterstützen (vgl. Tabelle 4.1). Die Organisationsstruktur bildet die *strategische Ebene* der Planung. In ihr werden die Grundstrukturen der bearbeitbaren Rollen, der Delegationspfade usw. festgelegt. Die Teamformation auf Basis der Prozessabwicklung des Organisationsnetzes stellen die *taktische Ebene* dar. Hier wird zu einer gegebenen Situation festgelegt, wie ein Team G zusammengestellt wird, das sich der Aufgabe annimmt. Die Auswahl eines Teamplanes, d.h. eines Dienstnetzprozesses π_G in Abhängigkeit zur organisationalen Steuerung ψ stellt die *operationale Ebene* dar. Hier wird entschieden, wie innerhalb des Teams Rollen in konkreten Aktionen ausgefüllt werden.

In Kapitel 2 haben wir bereits festgestellt, dass Rollenkomponenten von Dienstnetzen eine Spezifikation der mit einer Rolle verbundenen Rechte und Pflichten darstellen. Organisationen definieren durch ihre Positionen in ganz ähnlicher Weise ebenfalls Rechte und Pflichten – diesmal aber für die Positionen, die man sich als Bündelung von Rollen vorstellen kann. Die Rechte einer Position definieren sich über die Delegationsmöglichkeiten. Ihre Pflichten sind über die Rollen $R(O)$ definiert, d.h. über diejenigen Rollen, welche die Position O ausfüllen muss.

Betrachten wir nun, wie ein SONAR-MAS seine eigene Struktur zur Laufzeit verändern kann.

5 Sozionisches Modell reflexiver, selbstorganisierter Koordinierung

Bislang haben wir Organisationen als kontextuelle Strukturen begriffen, welche die Prozesse des Multiagentensystems rahmen. In diesem Abschnitt betrachten wir Organisationen als Agierende. Während sich bislang Prozesse eines Multiagentensystems als Reaktion auf externe Situationen darstellten, betrachten wir nun Reaktionen auf interne Situationen. Diese Prozesse der Organisation haben demnach sich selbst zum Gegenstand, sind also reflexiv.

5.1 Reorganisation als organisationale Aktivität

Wie unterscheiden Organisationsprozesse erster und zweiter Ordnung. Die Organisationsprozesse *erster Ordnung* betreffen das Zusammenspiel des Teams (engl. team work). Soll eine Aufgabe realisiert werden, so wird ein Team generiert, und jeder Positionsagent, der Teil des Teams ist, führt seinen Teil des gemeinsamen Teamplans aus. Teamplanung ist ein Organisationsprozess, der die Organisation – zumindest für den Moment der Teamgenese – als statisches Gebilde auffasst. Die Organisation ist der Kontext, innerhalb dessen sich Teamprozesse vollziehen.

Für Organisationsprozesse *zweiter Ordnung* ist die Organisation dagegen selbst Gegenstand des Prozesses. Es handelt sich um Reorganisationprozesse, die eine Organisation transformieren. Diese autopoetischen Reorganisationprozesse stößt die Organisation aufgrund interner Zustände an. Für Organisationsprozesse zweiter Ordnung sind Organisationen also die Variable. Organisationsprozesse zweiter Ordnung sind also, indem sie sich mit sich selbst beschäftigen, reflexiv.

An dieser Stelle kann man den Akteurscharakter der Organisation herausstellen, denn indem die Organisation reflexiv auf ihre eigenen Zustände Bezug nimmt und diese verändert, tritt sie als Akteur auf. Da die Reorganisation sich regelhaft auf interne Zustände bezieht, kann man in diesem Zusammenhang sogar von der Lernfähigkeit einer Organisation sprechen.

Von besonderer Bedeutung ist hierbei, dass die Transformationsprozesse nicht als unmittelbares Ergebnis der Aktivitäten der Organisationsmitglieder zu verstehen sind, denn die Organisation reagiert auf ihre internen Zustandsveränderungen, nicht aber auf das Verhalten der Organisationsmitglieder, die ursächlich die Zustandsveränderungen herbeigeführt haben. Die Organisation ist sogar als autonomer Akteur zu verstehen, denn die Reaktion auf Zustandsveränderungen erfolgt nach eigenen Regeln. Kurzgefasst: Die Organisation ist operational geschlossen.

Die Organisationsprozesse erster und zweiter Ordnung sind nicht unabhängig, sondern vielmehr wechselseitig aufeinander bezogen. Die Organisation tritt sowohl als Koordinator als auch als Koordinierter auf. Bei jedem Koordinierungsprozess sind sowohl Koordinator als auch Koordinierte einem wechselseitigem Anpassungsprozess unterworfen. Wir kennen diese Gedankenfigur allgemein auch für soziale Systeme, genauer für die wechselseitige Verflechtung der Akteure mir der sozialen

Struktur, bei denen jede Struktur mit einer Kontrolle einhergeht. Diese Kontrolle dient den Akteuren der Sicherung ihres eingebrachten Kapitals, ihrem Investionswert, und dies wird meist von Anerkennungs- und Sanktionierungsmaßnahmen flankiert.

Ein einfaches Beispiel für den reflexiven Charakter der Organisationsprozesse ist die Generierung einer modifizierten Steuerung ψ als Ergebnis einer Gruppeninteraktion (vergleiche dazu Abbildung 5.1). Die Organisation beeinflusst zum einen die Akteure, denn sie rahmt als Koordinator mit der Steuerung ψ den Plan jedes Teammitglieds, denn jedes Mitglied $i \in \{1, \ldots, n\}$ muss einen Plan wählen, der die Steuerungsbedingung erfüllt. Die Auswahl des Teams hat Auswirkungen auf die Organisationsmitglieder, da diese ihre Strukturen gegebenenfalls so reorganisieren, dass sie zu den Steuerungsbedingungen passen.[1] Es gilt zudem noch, dass sich das Team G auf einen gemeinsamen Teamplan π_G, d.h. ein Prozess des Teamnetzes $D(G)$, einigen muss, der die globale Steuerbedingung $\hat{\psi}(G)$ des Teams erfüllt. Außerdem muss der Teamplan π_G einen korrekten Ablauf des Dienstes beschreiben, d.h. er muss auch teminieren, ohne dass im Dienstnetz noch Marken verbleiben (vgl. Definition 44). Zum anderen beeinflussen die Akteure aber auch die Organisation, indem jene bewirken, dass sie die Steuerungsbedingungen anpasst. Dies ist notwendig, damit die Einigung eines Teams auf einen gemeinsamen Plan nicht zu aufwendig wird. Um den Suchraum zu begrenzen ist es wünschenswert, dass die Steuerungsbedingungen so beschaffen sind, dass sie die korrekte Termination implizieren. Dabei bietet es sich für die Organisation an, bei der Modifikation der Steuerung $\psi \rightsquigarrow \psi'$ an der Teamplanung zu orientieren, d.h. die Steuerung so zu modifizieren, dass sie mit den Konfliktvermeidungsstrategien der Organisationsmitglieder zusammenfällt.

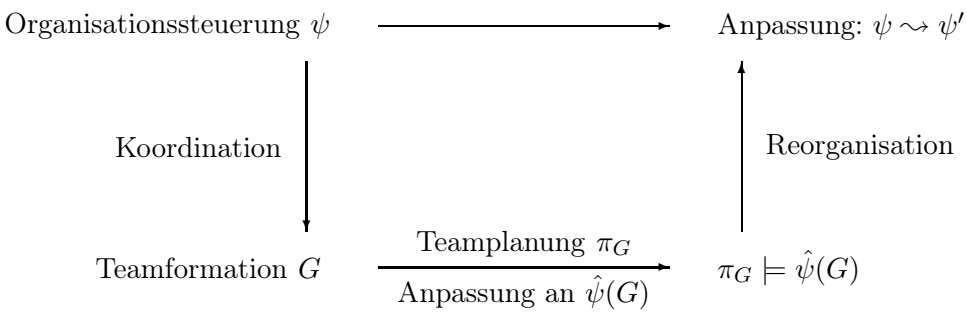


Abbildung 5.1: Rekursive Koordination

Auf diese Art und Weise generieren die Handlungen der Organisationsmitglieder die Organisationstrukturen, die später als Steuerungsbedingungen auf sie einwirken. Es gilt also, dass sich sowohl die Organisationsmitglieder als auch die Organisation, repräsentiert durch ihre Positionsagenten, *wechselseitig* einander anpassen. Man beachte, dass sich die wechselseitige Beeinflussung von Mitgliedern und Organisation in unserem Modell zu einer gewöhlichen Mitgliedsagent/Positionsagent-Relation, also zu einer Agent/Agent-Beziehung reduziert, da die Akteure und die Organisation als Ausprägungen des gleichen Konzeptes gesehen werden.

Wie in Kapitel **??** diskutiert ist der Unterschied zwischen Mikro- und Makroelementen eher inhaltlicher Natur, denn ihrer Struktur nach sind sie weitesgehend identisch. Die strukturelle Ähnlichkeit erlaubt es uns, einen neuen Blick auf die Modelle einzunehmen, bei dem Positionsagenten und „normale“ Agenten konzeptionell

[1] Zum Verhältnis von Organisation und Organisationsmitglied siehe Kapitel 6.

nicht unterschieden werden. Diese Vereinheitlichung hilft uns, Konzeptlücken zu schließen, die von den jeweiligen Metaphern bedingt sind. Die Metapher des Agenten impliziert eine Betonung des Handelns, des Planens etc. Sie verschließt sich dagegen tendenziel strukturellen Ansätzen. Umgekehrt impliziert die Metapher der Organisation Statik, die blind gegenüber möglichen Veränderung ist.

Eine Analyse der Gemeinsamkeiten zeigt, dass beide Sichtweisen ein gemeinsames Modell zugrundeliegt, das die aktionististischen wie die strukturellen Elemente gleichermaßen enthält. Wir erkennen, dass es sich bei diesem Vorhaben um die konstruktivistische Wendung der soziologischen Betrachtung aus Kapitel **??** handelt. Die rekursive Form des Entitätenmodells korrespondiert zum rekursiven Bezug der SONAR-Agenten, die sowohl Mikro- als auch Makrocharakter besitzen.

Pointiert formuliert handelt es sich bei den Organisationsprozesse erster Ordnung um *selbstorganisierte Koordinierungsprozesse* und bei denen zweiter Ordnung um *Prozesse koordinierter Selbstorganisation*. Die Mikro-/Makro-Dualität zeigt sich hier im Akzent: Im Falle selbstorganisierte Koordinierung betrachten wir den Koordinierungsprozess aus der Perspektive der koordinierten Einheiten, im Falle koordinierter Selbstorganisation aus der Perspektive der koordinierenden Einheit.

Interessante Konstellationen sind gerade solche, bei denen die Anpassung der einen Seite vernachlässigt werden kann. Interessant sind hier also die beiden Extreme. Wird beispielsweise innerhalb des Prozesses der Koordinator erst generiert, so ist diese Dynamik von größerem Interesse als die daraus resultierende Koordinierung. Wird dagegen ein bestehendes Protokoll eingesetzt, um den Prozess zu koordinieren, so kann die Koordinationsstruktur näherungsweise als statisch angesehen werden, und die koordinierten Akteure rücken in den Mittelpunkt.

5.2 Reorganisation durch Organisationstransformationen

Im folgenden konzeptionalisieren wir Reorganisationsprozesse als Transformationen, die eine formale Organisation auf eine andere abbildet. In diesem Kapitel wollen wir zunächst die Organisation als ganzes betrachten, d.h. wir gehen davon aus, dass wir eine globale Sicht auf das System besitzen, um Reorganisationen vorzunehmen. Eine lokale Sichtweise, bei der die Modifikation die Organisationsstruktur durch Anpassung der Positionsagenten erfolgt, ist Gegenstand des nachfolgenden Kapitels 5.3.

Wir betrachten nun Transformation auf Organisationen. Eine Transformation bildet eine MAS-Organisation auf eine weitere ab. Die Transformation bezieht sich im allgemeinen auf das Organisationsnetz N, auf die Organisationsstruktur $\mathcal{O}$, auf die Rollenzuweisung $R : P \to \mathcal{R}$, auf die Dienstnetzzuweisung $D : T \to \mathcal{D}$, auf die Prozesssteuerung ψ und auf die Agentenzuweisung $Ag : \mathcal{O} \to \mathcal{S}$:

$$(N_1, \mathcal{O}_1, R_1, D_1, \psi_1, Ag_1) \mapsto (N_2, \mathcal{O}_2, R_2, D_2, \psi_2, Ag_2)$$

Transformationen heißen nebenläufig, wenn sie sich auf disjunkte Teile der Organisation beziehen. Wenn die Transformationen f_1 und f_2 nebenläufig sind, dann ist $f_1 \oplus f_2$ eine Transformation, die den gemeinsamen Effekt der Einzeltransformationen beschreibt.

Definition 53 *Zwei Transformationen f_1 und f_2 heißen nebenläufig, wenn gilt:*

$$(f_1(x) \neq x \Longrightarrow f_2(x) = x) \wedge (f_2(x) \neq x \Longrightarrow f_1(x) = x)$$

Die Summe $f_1 \oplus f_2$ zweier nebenläufiger Transformation ist folgendermaßen definiert:

$$(f_1 \oplus f_2)(x) = \begin{cases} f_1(x), & \textit{falls } f_1(x) \neq x, f_2(x) = x \\ f_2(x), & \textit{falls } f_1(x) = x, f_2(x) \neq x \\ x, & \textit{sonst} \end{cases}$$

Mit $\mathcal{ORG}$ bezeichnen wir die Menge aller Organisationen. Wir nehmen an, dass die Stellenmenge P aller Organisationen in $\mathcal{ORG}$ Teilmenge einer universellen Stellenmenge $\mathcal{P}_u$ ist und dass die Transitionsmenge T Teilmenge einer universellen Menge $\mathcal{T}_u$ ist. Damit vermeiden wir, dass ein Element x in einem Organisationsnetz eine Stelle beschreibt, in einem anderen dagegen eine Transition. Die hier eingeführte Konvention ist daher lediglich eine Typisierung und keine prinzipielle Einschränkung. Eine Transformation f ist dann eine Abbildung von der Menge $\mathcal{ORG}$ auf sich selbst:

$$f : \mathcal{ORG} \rightarrow \mathcal{ORG}$$

Mit $\mathcal{TM}$ bezeichnen wir die Menge aller Transformationen.

Transformationen sind verkettbar, d.h. wenn f_1 und f_2 zwei Transformationen sind, dann ist auch $f_1(f_2(Org, Ag))$ eine. Mit der identischen Transformation haben wir damit die Kategorie der Organisationstransformation mit den Organisationen $\mathcal{ORG}$ als Objekten und den Transformationen $\mathcal{TM}$ als Morphismen.

Wir betrachten nur *wohlgeformte Transformationen*, d.h. Transformationen auf Organisationen, die eingeschränkt auf wohlgeformte Organisationen einen Abschlußoperator darstellen, d.h. die eine wohlgeformte Organisation auf eine solche abbilden.

5.3 Selbstorganisation durch Transformationsteams

Betrachten wir zunächst, wie sich das Verhältnis von Organisation und generierten SONAR-MAS sich durch eine Transformation verändert. Eine Reorganisation durch eine Organisationstransformation ist konzeptionell zentralisiert, denn wenn wir die Transformation $(Org_1, Ag_1) \mapsto (Org_2, Ag_2)$ vornehmen, dann verändert sich auch das davon generierte SONAR-MAS von $MAS_1 = \mu(Org_1, Ag_1)$ zu $MAS_2 = \mu(Org_2, Ag_2)$. Es ist also notwendig, ein globales Modell der Organisation zu verwalten, das transformiert wird und von dem sich dann das veränderte Multiagentensystem ableiten lässt (vgl. dazu die obere, rechte Hälfte der Abbildung 5.2).

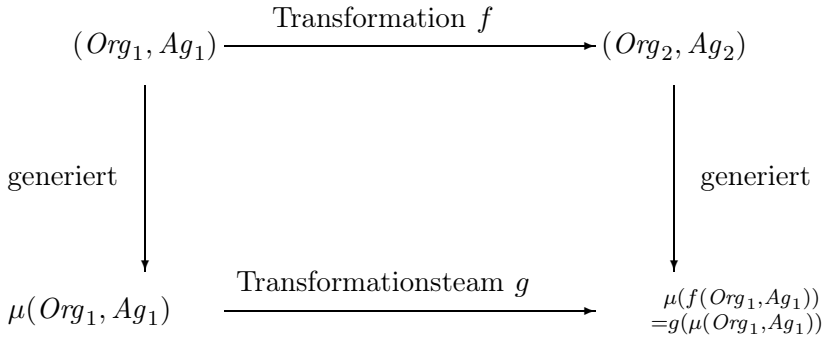


Abbildung 5.2: Zentrale versus dezentrale Transformation

In diesem Abschnitt entwickeln wir erweiternd eine denzentralisierte Perspektive auf die Organisationstransformation. Die Organisation (Org_1, Ag_1) generiert initial das SONAR-MAS $MAS_1 = \mu(Org_1, Ag_1)$. Die Organisationsstruktur liegt dann verteilt im Multiagentensystem vor. Um eine denzentralen Transformation vorzunehmen wird diese Organisationsstruktur nicht rekonstruiert – stattdessen transformieren die Agenten des SONAR-MAS MAS_1, so dass ein modifiziertes System $MAS_2 = g(MAS_1)$ generiert wird. Diese Transformation findet kooperativ in einem *Transformationsteam* statt (vgl. dazu die untere, linke Hälfte der Abbildung 5.2).

Ziel ist es dabei, dass die beteiligten Agenten ihre Präferenzen so anpassen, dass die *lokalen* Anpassungen auf der Ebene des SONAR-MAS das gleiche Ergebnis haben wie die *globale* Transformation auf Ebene der Organisation. Formal heißt dies, dass das Diagramm in Abbildung 5.2 kommutieren soll:

$$\mu(f(Org_1, Ag_1)) = g(\mu(Org_1, Ag_1))$$

In Kurzform: $\mu \circ f = g \circ \mu$. Betrachten wir nun, wie diese Lokalisierung zu erreichen ist.

5.3.1 Lokale Transformationen

Eine Transformation ist lokal, wenn sich die Transformation nur auf eine Organisationsposition O bezieht, d.h. wenn sich im Kontext $\bar{\mathcal{O}} := \mathcal{O}_1 \setminus \{O\}$ keine Veränderungen ergeben. Sie ist übergreifend, wenn die Änderung an einer Position auch welche an anderen Positionen nach sich zieht.

Definition 54 *Sei $O \in \mathcal{O}$ eine Position. Eine Transformation λ ist* O-lokal*, wenn folgendes für jedes Bild von λ gilt:*

$$\lambda : (N_1, \mathcal{O}_1, R_1, D_1, \psi_1, Ag_1) \mapsto (N_2, \mathcal{O}_2, R_2, D_2, \psi_2, Ag_2)$$

1. *Die Position des Kontextes bleiben erhalten:*

$$(\mathcal{O}_1 \setminus \{O\}) \subseteq \mathcal{O}_2$$

2. *Im Kontext beginnende Kanten bleiben erhalten:*

$$((x, y) \in F_1 \wedge x \in \bar{\mathcal{O}}) \Longrightarrow (x, y) \in F_2$$

3. *Die Anschriften bleiben im Kontext erhalten:*

$$\begin{aligned} R_1|_{(P_1 \cap \bigcup \bar{\mathcal{O}})} &= R_2|_{(P_2 \cap \bigcup \bar{\mathcal{O}})} \\ D_1|_{(T_1 \cap \bigcup \bar{\mathcal{O}})} &= D_2|_{(T_2 \cap \bigcup \bar{\mathcal{O}})} \\ \psi_1|_{(T_1 \cap \bigcup \bar{\mathcal{O}})} &= \psi_2|_{(T_2 \cap \bigcup \bar{\mathcal{O}})} \\ Ag_1|_{(\bigcup \bar{\mathcal{O}})} &= Ag_2|_{(\bigcup \bar{\mathcal{O}})} \end{aligned}$$

Eine Transformation heißt lokal, wenn sie O-lokal für ein O ist.
Eine Transformation f heißt lokalisierbar*, wenn sie sich als Komposition lokaler Transformation darstellen lässt.*

Bei einer O-lokale Transformationen verändert sich die Zusammensetzung von O, beispielsweise, indem Knoten gelöscht oder hinzugefügt werden. Trotzdem kann eine andere Menge O' den Platz von O gewissermaßen ersetzen. Man beachte aber, dass

O-lokale Transformationen nicht implizieren, dass O stets in diesem Sinne erhalten bleibt, denn O kann sogar als ganzes gelöscht werden oder durch mehrere Positionen $O_1, \ldots, O_n$ ersetzt werden. Eine O-lokale Transformation schließt mit ein, dass der Positionsagent $Ag(O)$ gelöscht oder auf mehrere aufgeteilt wird.

Lemma 9 *Sei λ eine O-lokale Transformation.*

- *Die Knoten im Kontext bleiben erhalten:*

$$(P_1 \cap \bigcup \bar{\mathcal{O}}) \subseteq P_2 \quad \textit{und} \quad (T_1 \cap \bigcup \bar{\mathcal{O}}) \subseteq T_2$$

- *Knoten, die mit dem Kontext verbunden sind, bleiben erhalten:*

$$((x,y) \in F_1 \wedge x \notin \bar{\mathcal{O}} \wedge y \in \bar{\mathcal{O}}) \Longrightarrow y \in \bigcup \mathcal{O} \setminus \bigcup \bar{\mathcal{O}}$$

- *Die Kanten des Kontextes bleiben erhalten:*

$$(F_1 \cap \bar{\mathcal{O}}^2) \subseteq F_2$$

- *Das Organisationsnetz verändert sich nicht im Kontext:*

$$N_1|_{\bar{\mathcal{O}}} = N_2|_{\bar{\mathcal{O}}}$$

Beweis: (1) Da die Position des Kontextes $\bar{\mathcal{O}}$ erhalten bleiben und da für jedes $O' \in \bar{\mathcal{O}}$ auch $O' \subseteq P_1 \cup T_2$ gilt, müssen auch alle Knoten in O' erhalten bleiben, d.h. $\bigcup \bar{\mathcal{O}} \subseteq (P_2 \cup T_2)$ gilt. Da wir gefordert haben, dass alle Stellenmengen Teilmenge der universellen Menge $\mathcal{P}_u$ sind (und analog auch für Transitionen), folgen die Inklusion sogar jeweils für Stellen und Transitionen.

(2) Wenn eine Kante $(x,y) \in F_1$ in O endet, aber nicht dort beginnt, dann folgt aus der Definition, dass die Kante auch im Bild von λ sein muss. Also müssen auch x und y im Bild sein, denn das Bild eines Organisationsnetzes unter λ ist wiederum ein Netz.

(3) Die Aussage ist äquivalent zu $(x,y) \in (F_1 \cap \bar{\mathcal{O}}^2) \Longrightarrow (x,y) \in F_2$ Die Kanten des Kontextes starten insbesondere dort, bleiben also nach Definition erhalten.

(4) Direkt aus (1) und (3). q.e.d.

Lokale Transformation entsprechen der Nebenläufigkeit.

Theorem 47 *Wenn $O_1, O_2 \in \mathcal{O}$ zwei verschiedene Positionen sind, dann sind die O_i-lokale Transformationen λ_i für $i = 1, 2$ nebenläufig.*

Beweis: Wenn beide Transformation für verschiedene Positionen lokalisiert sind, dann werden auch unterschiedliche Netzknoten modifiziert, womit die Nebenläufigkeit folgt. q.e.d.

Manche Transformationen lassen sich nicht mit einer lokalen Transformation erledigen. Aus der Definition einer lokalen Transformation folgt, dass keine Kante (p,t) entfernt werden kann, wenn sie außerhalb, d.h. im Kontext von O startet, d.h. wenn $p \in \bigcup \bar{\mathcal{O}}$ gilt. Inhaltlich bedeutet dies, dass keine Positionsagent eine Delegationskante löschen darf. Erst recht darf eine Position keine Transition t löschen. Will eine Position die Transition t löschen, muss sie die Position, in der die Delegationskante (p,t) startet, überzeugen, diese Kante zu löschen. Wenn diese dann nicht mehr existiert, dann ist O frei, die Transition t selbst zu entfernen, da t nicht vernetzt ist. Abbildung 5.3 illustriert das Verfahren. Dieses koordinierte Handeln zur Transformation bringt uns im folgenden zum Konzept der Transformationsteams.

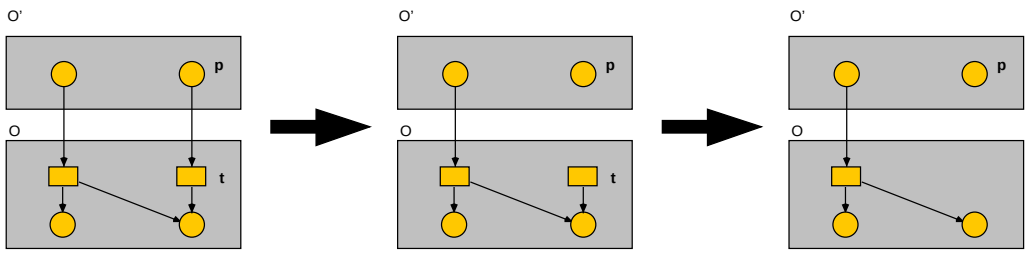


Abbildung 5.3: Koordinierte Transformation

5.3.2 Transformationsteams

Wir verwenden Transformationsteam, um eine agentenbasierte Komposition von Transformationen beschreiben zu können. Jeder Transition im Team wird dazu eine Transformation zugeordnet.

Definition 55 *Ein* Transformationsteam $TT = (N, R, D, \psi, \lambda)$ *besteht aus den folgenden Komponenten:*

1. (N, R, D, ψ) *ist ein koordinierendes Team.*
2. $\lambda : T \to \mathcal{TM}$ *ist die Transformationsanschrift.*

Wir interessieren uns im folgenden für solche Teams, bei denen alle $\lambda(t)$ lokal sind. Transformationsteams lassen sich dann nur in Zusammenhang mit Organisationen definieren, denn dann müssen wir wissen, welche Positionen O zur Verfügung stehen. Wir erweitern Organisationen zunächst zu transformierende Organisationen, indem wir jeder Transition t die Transformation $\lambda(t)$ zuweisen. Dies soll bedeuten, dass wann immer die Transition t schaltet, die Organisation die Transformation $\lambda(t)$ vornimmt.

Definition 56 *Eine* transformierende Organisation

$$(N, \mathcal{O}, R, D, \psi, \lambda)$$

besteht aus einer Organisation $(N, \mathcal{O}, R, D, \psi)$ *und einer Transformationsanschrift* $\lambda : T \to \mathcal{TM}$*, so dass für alle* $O \in \mathcal{O}$ *gilt, dass alle* $\lambda(t)$ *für* $t \in (T \cap O)$ *jeweils* O*-lokal sind.*

Eine transformierende Organisation heißt wohlgeformt*, falls* $(N, \mathcal{O}, R, D, \psi)$ *dies ist und für alle Positionen* $O \in \mathcal{O}$ *gilt, dass die Transformationen* $\lambda(t)$ *und* $\lambda(t')$ *für alle* $t, t' \in (T \cap O)$ *mit* $t \neq t'$ *paarweise nebenläufig sind.*

Jede Organisation kann als eine transformierende betrachtet werden, wenn wir jeder Transition die Identitätstranformation, die keinerlei Modifikationen vornimmt, zuweisen: $\lambda(t) = id$. Offensichtlich verändert sich dadurch nichts an den bislang beschriebenen Eigenschaften einer Organisation.

Wir betrachten nun, inwieweit sich die bestehenden Mechanismen der Teamformation nutzen lassen, um die Bildung von Transformationsteams zu unterstützen. Wir wissen aus Lemma 34, dass jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ einer Organisation *Org* das Team $G_{Org}(K, \phi)$ generiert. Völlig analog gilt dies auch für Transformationsteams.

Lemma 10 *Sei Org eine transformierende Organisation und $R \in \mathcal{R}$ eine Rolle. Für jedes $p \in P_{Org}(R)$ erzeugt jeder Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ des Organisationsnetzes das Transformationsteam:*

$$TT_{Org}(K, \phi) := (K, (R \circ \phi), (D \circ \phi), (\psi \circ \phi_{|E}), (\lambda \circ \phi_{|E}))$$

Beweis: Wir wissen bereits aus Lemma 34, dass $(K, (R \circ \phi), (D \circ \phi), (\psi \circ \phi_{|E}))$ ein koordinierendes Team ist. Offensichtlich ist $\lambda \circ \phi_{|E}$ eine Transformationsanschrift. q.e.d.

Da wir nur an dem gesamten Team, nicht aber an den Einzeltransformation interessiert sind, ist es nicht notwendig, dass schon die Transformationen $\lambda(t)$ wohlgeformt sind. Es ist jedoch wünschenswert, dass das gemeinschaftliche Werk eines Teams insgesamt eine wohlgeformte Transformation beschreibt.

Wir definieren nun die Transformation, die von einem Team beschrieben wird. Sei $TT = (N, R, D, \psi, \lambda)$ ein Transformationsteam. Mit Hilfe eines Transformationsteam können wir rekursiv eine Transformation $\lambda(TT)$ definieren, nämlich dann, wenn für alle $p \in t^\bullet$ die Subteams $TT_{\Downarrow p}$ ihrerseits Transformationen $\lambda(TT_{\Downarrow p})$ darstellen, die bezüglich $\lambda(t)(Org)$ paarweise nebenläufig zueinander sind. In diesem Fall ist die Operatorensumme $\bigoplus_{p \in t^\bullet} \lambda(TT_{\Downarrow p})$ definiert, und wir können für das Team TT die folgende Transformation festlegen:

$$\lambda(TT) := \left(\bigoplus_{p \in t^\bullet} \lambda(TT_{\Downarrow p}) \right) \circ \lambda(t) \quad \text{mit} \quad {}^\circ TT = \{p_0\}, p_0{}^\bullet = \{t\} \tag{5.1}$$

Jedes Teamnetz hat nur eine initiale Stelle, d.h. es gibt stets ein p_0, so dass ${}^\circ TT = \{p_0\}$ gilt. Außerdem hat jede Stelle im Team genau eine Transition im Nachbereich, d.h. $p_0{}^\bullet = \{t\}$ für ein t. Also ist die obige Darstellung eindeutig.

Man beachte, dass dieser Ausdruck nur unter den obigen Bedingungen der Nebenläufigkeit definiert ist. Diese sind im allgemeinen schwer zu garantieren. Generieren wir die Transformationsteams, die eine Organisation transformieren sollen, durch die Organisation selbst, so können wir dies leichter sicherstellen. Wir betrachten speziell wohlgeformte, transformierende Organisationen.

Theorem 48 *Wenn $(N, \mathcal{O}, R, D, \psi, \lambda)$ eine wohlgeformte, transformierende Organisation ist, dann beschreibt das Transformationsteam $TT_{(Org,\psi)}(K, \phi)$ für jede Rolle $R \in \mathcal{R}$ und jede $p \in P_{Org}(R)$ eine Transformation, d.h.*

$$\lambda(TT_{Org}(K, \phi))$$

nach (5.1) ist eine wohldefinierte Transformation.

Beweis: Wir wissen bereits aus Theorem 47, dass die Transformationen verschiedener Positionen zueinander nebenläufig sind. In wohlgeformten transformierenden Organisation sind zudem noch die Transformationen $\lambda(t)$ einer Position zueinander nebenläufig. Also ist die Summe $\bigoplus_{p \in t^\bullet} \lambda(TT_{\Downarrow p})$ für alle Subteams definiert und damit auch $\lambda(TT)$. q.e.d.

Wir sehen, dass die Bedingung der wohlgeformten, transformierenden Organisation, nämlich dass alle Transformationen $\lambda(t)$ einer Position zueinander nebenläufig sein müssen, strenger ist, als wir es in dieser Aussage benötigen. Es wäre für den

obigen Beweis ausreichend, die Nebenläufigkeit für alle Transformationen zu fordern, die tatsächlich in Prozessen gleichzeitig vorkommen können. Diese Menge können wir bestimmen, indem wir die mit der Organisation assoziierte kontextfreie Grammatik analysieren. Wir führen die dazu notwendigen Definitionen nicht aus, da sie ganz analog zu den bisherigen Konstruktionen verlaufen und keine neuen Erkenntnisse bringen.

5.3.3 Transparente Transformationen

Wir betrachten nun die Operationalisierung, d.h. die Ausführung der lokalen Transformationen im Multiagentensystem, wie wir es einleitend im Diagramm 5.2 beschrieben haben.

Wie wir gesehen haben, impliziert eine O-lokale Transformationen, dass der Kontext von O unverändert bleibt. Nur O verändert sich. Man könnte daher glauben, dass beim parallel stattfindenden Übergang auf der Ebene des Multiagentensystems:

$$\mu(N_1, \mathcal{O}_1, R_1, D_1, \psi_1, Ag_1) \mapsto \mu(N_2, \mathcal{O}_2, R_2, D_2, \psi_2, Ag_2)$$

ebenfalls alle Positionsagenten im Kontext erhalten bleiben. Dies ist aber leider nicht so: Nehmen wir an, dass O bislang lediglich aus den beiden Transitionen t_1 und t_2 bestand und dass eine Transformationen O in die zwei Positionen $O_1 = \{t_1\}$ und $O_2 = \{t_2\}$ aufgespalten wird (vgl. Abbildung 5.4). Diese Transformation ist O-lokal. Jede andere Position O', die eine Delegationskante von der Form (p, t_1) bzw. (p, t_2) besitzt, hat in den Handlungsmöglichkeiten $\Delta_{Ag(O')}$ einen Eintrag, in dem $Ag(O)$ als Delegationspartner eingetragen ist. Nach der Transformation stehen aber nur die beiden Positionsagenten $Ag(O_1)$ und $Ag(O_2)$ als Delegationspartner zur Verfügung. Die Modifikation muss sich also auch in $Ag(O')$ widerzuspiegeln. Die Transformation ist somit lokal auf der Ebene der Organisation, aber nicht auf Ebene der Positionsagenten.

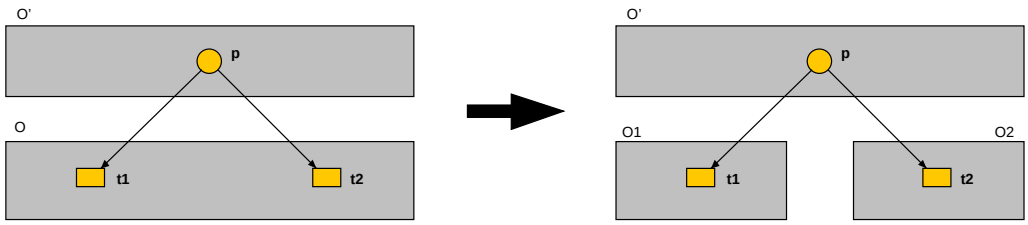


Abbildung 5.4: Lokale Transformation einer Position

Wir fragen uns nun, welche O-lokalen Transformationen nur durch eine Modifikation des Positionsagenten $Ag(O)$ allein durchgeführt werden können.

Hierbei sind alle externen Referenzen auf O zu erhalten. Sei

$$\lambda : (N_1, \mathcal{O}_1, R_1, D_1, \psi_1, Ag_1) \mapsto (N_2, \mathcal{O}_2, R_2, D_2, \psi_2, Ag_2)$$

eine O-lokale Transformation mit $O \in \mathcal{O}_1$ und sei $|\mathcal{O}_1| = |\mathcal{O}_2|$. Da die Positionen des Kontextes erhalten bleiben, d.h. $(\mathcal{O}_1 \setminus \{O\}) \subseteq \mathcal{O}_2$ gilt folgt aus $|\mathcal{O}_1| = |\mathcal{O}_2|$, dass es ein O_2 gibt, so dass

$$(\mathcal{O}_1 \setminus \{O\}) \uplus \{O_2\} = \mathcal{O}_2 \quad \text{bzw.} \quad \mathcal{O}_2 \setminus (\mathcal{O}_1 \setminus \{O\}) = \{O_2\}$$

Gilt für diese Positionen

$$Ag(O) = Ag(O_2),$$

dann geschieht die Ersetzung von O durch O_2 transparent für Positionsagenten im Kontext.

Definition 57 *Eine O-lokale Transformation λ mit $O \in \mathcal{O}_1$ heißt* O-transparent*, falls gilt:*

1. $|\mathcal{O}_1| = |\mathcal{O}_2|$
2. $Ag(O) = Ag(O_2)$ *für* $\{O_2\} = \mathcal{O}_2 \setminus (\mathcal{O}_1 \setminus \{O\})$.

Um die Transformation auch tatsächlich operationalisieren zu können, wollen wir natürlich, dass der Teamagent $A(t)$, der die O-transparente Transformation $\lambda(t)$ vornimmt, identisch ist mit dem Positionsagenten $Ag(O(t))$, zu dem t gehört:

$$t \in (T \cap O) \Longrightarrow \lambda(t) \text{ ist } O\text{-transparent} \tag{5.2}$$

Die Äquivalenz kann nicht stets gelten, da beispielsweise die Identität für jedes beliebige O auch O-transparent ist.

Betrachten wir daher wohlgeformte, transformierende Organisationen, bei denen die Transformationen $\lambda(t)$ nicht nur O-lokal, sondern sogar O-transparent sind. Solche Organisationen nennen wir transparent.

Theorem 49 *Sei Org eine transparente Organisation. Für jedes durch einen Organisationsprozess generiert Transformationsteam $TT_{Org}(K, \phi)$ gilt:*

$$\forall e \in E : \phi(e) \in (T \cap O) \Longrightarrow \lambda(\phi(e)) \textit{ ist } O\textit{-transparent}$$

Beweis: Nach der Definition von λ in Def. 56 gilt, dass alle $\lambda(t)$ für $t \in (T \cap O)$ jeweils O-lokal sind. Betrachten wir nur eine transparente Organisationen, so ist $\lambda(t)$ nicht nur O-lokal, sondern sogar O-transparent, und für $t = \phi(e)$ folgt dann die Aussage. q.e.d.

Jede O-transparente Transformation kann vom Positionsagenten $Ag(O)$ allein durchgeführt werden.

Theorem 50 *Wenn λ eine O-transparente Transformation $(Org_1, Ag_1) \mapsto (Org_2, Ag_2)$ ist, dann bleiben in den Bildern $\mu(Org_1, Ag_1)$ und $\mu(Org_2, Ag_2)$ die Agentennamen des Multiagentensystems unverändert. Die Interna des Positionsagenten $Ag(O)$ verändern sich, die Interna aller anderen bleiben unverändert.*

Beweis: Man beachte, dass $Ag_i, i = 1, 2$ nach Definition eine Injektion ist. Jede Position O ist eindeutig dem Positionsagenten $Ag_1(O)$ zugeordnet. Wir betrachten dazu die Konstruktion der Agenten $Ag(O)$ von $\mu(Org_i, Ag_i)$ nach Definition 52. Da λ transparent ist, gilt $Ag_1(O) = Ag_2(O)$. Sei nun $A_O := Ag_1(O)$.

1. Die Rollenfähigkeiten

   $$\rho_{A_O}(D_0) = \{R(t^\bullet) \mid t \in Ag^{-1}(A_O) \cap T, D(t) = D_0\}$$

   sind lokal definiert, da alle $t \in Ag^{-1}(A_O)$ und alle Stellen $p \in t^\bullet$ zu einer Position O gehören. Daher bezieht sich die Definition ρ_{A_O} nur auf eine Position.

2. Die Teamgeneration $\gamma_{A_O}(R) = G(K_{\tau_{A_O}}(N, R))$ ist O-lokal, da $\tau^3_{A_O}$ dies ist.

3. Die Handlungsmöglichkeiten

$$\Delta_{A_O}(G)(b) = \{ \quad (D(t), \{(R(p), A_p) \mid p \in t^\bullet\}) \quad \mid \\ t \in (T \cap \phi(b)^\bullet \cap Ag^{-1}(A_O)), \forall p \in t^\bullet : A_p \in \mathcal{A}_{dlg}(p) \}$$

   sind nicht lokal, da sich die Definition nicht nur auf die Knoten t und $t^\bullet$ einer Position O – also die lokalen Elemente – , sondern auch auf die Agenten $\mathcal{A}_{dlg}(p) = \{Ag(O(t)) \mid t \in p^\bullet\}$ nach (4.5), also auch auf andere Positionen, bezieht. Da λ sogar O-transparent ist und somit $Ag(O) = Ag(O_2)$ gilt, verändert sich jedoch $\mathcal{A}_{dlg}(p)$ nicht, und die Modifikation hat doch nur lokale Auswirkung.

4. Die Handlungsmöglichkeiten $\delta_{A_O}(G)(\Delta)(b)$ ergeben sich anhand des Teams G – sind also unabhängig von den Positionen.

5. Die Definition des Handlungsplans $\pi_{A_O}(G)(D)(p) \in Proc(D(G), \hat{\psi}(G))$ ist ebenfalls unabhängig von den Positionen.

Also sind alle Modifikation unabhängig von der Positionsstruktur oder die Auswirkungen lokal begrenzt. Da eine O-lokale Transformation nur Änderungen innerhalb von O vornimmt, ändert sich daher nur A_O und alle anderen Positionsagenten bleiben unverändert, q.e.d.

Wir verwenden also Transformationsteams, um die Transformation f als koordinierte Komposition von O-transparenten Transformationen beschreiben zu können. Sind alle Transformationen transparent, dann lässt sich nach Theorem 50 die Transformation durch lokale Modifikation an den Positionsagenten vornehmen.

5.3.4 Lokalisierte Transformationsteams mit Delegationsabschluß

Wenn die Transformation $\lambda(t)$ nur O-lokal, aber nicht transparent ist, dann wird bei der Transformationen die Position O in n Positionen $O_1, \ldots, O_n$ aufgespalten. Ein Beispiel für $n = 2$ zeigt Abbildung 5.4. Die Transformation kann von $Ag(O)$ realisiert werden, indem der Agent $Ag(O)$ zunächst n neue Agenten $A_1, \ldots, A_n$ mit $Ag(O_i) := A_i$ für alle i erzeugt, diese mit den von $\lambda(t)$ beschriebenen Interna ausstattet und sich anschließend selbst löscht. Hierbei ist auch $n = 0$ möglich, was bedeutet, dass der Agent sich nur selbst löscht.

Wie wir im Beweis von Theorem 50 gesehen haben, sind von der Transformation alle Positionsagenten A betroffen, die potentiell an $Ag(O)$ delegieren, sobald die Transformation $\lambda(t)$ nur O-lokal, aber nicht transparent sind, denn dann ist in Δ_A ein Verweis auf $Ag(O)$ gespeichert. Damit die Transformation von den Agenten realisiert werden kann, müssen wir also alle Agenten informieren, die an O delegieren. Dies können wir direkt am Organisationsnetz ablesen, denn eine Delegation von O_1 an O_2 entspricht genau einer Kante (p, t) mit $O_1 = O(p)$ und $O_2 = O(t)$. Die Menge der Positionen, die an O delegieren, ist nach (4.6) die Menge $\mathcal{O}_{gen}(t)$.

Wir können sicher sein, dass die nutzenden Positionen ihre internen Datenstrukturen (d.h. $\Delta_{Ag(O)}$) an die Veränderung anpassen, wenn wir alle Agenten in der Menge $\mathcal{A}_{gen}(t)$ benachrichtigen. Um dies sicherzustellen, fordern wir, dass für jedes Ereignis e im Teamnetz und zu jeder an $O(\phi(e))$ delegierende Position $O_g \in \mathcal{O}_{gen}(\phi(e))$ mindestens ein Netzknoten $n \in B_K \cup E_K$ vorhanden ist, dessen Bild $\phi(n)$ in O_g liegt.

Theorem 51 *Ein Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ einer wohlgeformten, transformierenden Organisation Org besitzt die Eigenschaft des Delegationsabschluß, wenn gilt:*

$$\forall e \in E_K : \forall O_g \in \mathcal{O}_{gen}(\phi(e)) : O(\phi(B_K \cup E_K)) \cap \{O_g\} \neq \emptyset$$

Für jeden Prozess $(K, \phi) \in \mathcal{K}^{pg}(N, \{p\})$ mit Delegationsabschluß kann die Teamtransformation

$$\lambda(TT_{Org}(K, \phi))$$

des generierten Teams $TT_{Org}(K, \phi)$ verteilt durch lokale Meta-Planung implementiert werden.

Beweis: Verglichen mit dem Beweis von Theorem 50 ist hier nur noch zu zeigen, dass an $Ag(O)$ delegierende Positionsagenten, d.h. alle $O \in \mathcal{O}_{gen}(t)$, an der Transformation teilnehmen, d.h. Teammitglieder sind, damit sie in ihren Präferenzordnungen Δ_A den Verweis auf $Ag(O)$ anpassen können. Genau dies ist aber durch den Delegationsabschluß bereits sichergestellt. Da alle Teams von der Organisation als Prozesse abgeleitet werden, impliziert dies, dass in jedem Transformationsteam $TT_{Org}(K, \phi)$ alle Positionsagenten involviert sind, die an den sich transformierenden Positionsagenten $Ag(O)$ delegieren. Unter dieser Annahme ist auch für O-lokale Transformation sichergestellt, dass die Transformation, die durch das Team beschrieben wird, auch wirklich verteilt implementiert werden kann. q.e.d.

Delegationsabschluß ist besonders leicht herzustellen, wenn er sogar strukturell gilt, d.h. wenn für jede Schaltfolge des Organisationsnetzes, die mit dem Nachbereich der Transition t schalten kann, und für jede Position $O \in \mathcal{O}_{gen}(t)$ mindestens eine Transition t_O aus O schalten muss:

$$\begin{aligned} \forall t \in T : \quad & \forall O \in \mathcal{O}_{gen}(t) : \forall w \in T^* : t^\bullet \xrightarrow{w} \mathbf{0} \\ & \Longrightarrow \exists t_O \in T \exists u, v \in T^* : w = u t_O v \wedge t_O \in O \end{aligned}$$

Diese Eigenschaft ist aber in praktischen Systemen im allgemeinen zu streng, als dass man sie strukturell fordern sollte.

5.3.5 Lokale Transformation als Meta-Planung

Jede O-transparente Transformation eines Transformationsteams kann nach Theorem 50 vom Positionsagenten $Ag(O)$ allein, d.h ohne Mithilfe anderer, durchgeführt werden. Diese lokalen Modifikationsind bereits im Modell des SONAR-Agenten enthalten. Sie werden in Form von *Meta-Planer* umgesetzt. Sie haben die Aufgabe, Planungsstrategien anpassen oder Handlungsroutinen erlernen usw. Im Modell aus Abbildung 3.9 sind dies die beiden Transitionen Interdependenzen bewältigen und inkorporieren. Wie wir bereits gesehen haben, erfolgt im SONAR-Modell die Meta-Planung der Interdependenzbewältigung durch eine Anpassung der Selektionsfunktion δ und π. Die Transition inkorporieren modifiziert mit Hilfe des Meta-Planers β^1 die restlichen Interna des Agenten:

$$\beta^1(\rho, \gamma, \Delta) = (\rho', \gamma', \Delta')$$

Im Beweis von Theorem 50 ist implizit schon enthalten, wie diese Modifikationen vorzunehmen sind. Die Inkorporation bezieht sich auf die Rollenfähigkeiten $\rho_{Ag(O)}(D)$, die Teamgeneration $\gamma_{Ag(O)}(R)$ und die Handlungsmöglichkeiten

$\Delta_{Ag(O)}(G)$, denn diese Agenteninterna korelieren direkt mit der Organisationsstruktur, während die restlichen Komponenten nur Auswahlen unter von diesen Komponenten bereitgestellten Optionen treffen.

Bezeichne $A_O = Ag_1(O) = Ag_2(O)$ den sich transparent transformierenden Positionsagenten.

Der Teamselektor $\tau^3_A(R) \in P_{Org}(R)$ ändert sich unter Umständen, da sich Menge der Profile $P_{Org}(R)$, die die Rolle R anstoßen können, ändert, wenn sich P_1 zu P_2 ändert, denn nach (3.4) ist $P_{i,RD}(R) = \{p \in P_i \mid R(p) = R, {}^\bullet p = \emptyset\}$ von $P_i, i = 1, 2$ abhängig. Bei der Teamtransformation ändert sich die Teamformation, da diese indirekt durch $\tau^3_{A_O}(R)$ definiert ist: $\gamma_{A_O}(R) = G(K_{\tau_{A_O}}(N, \tau^3_{A_O}(R)))$ Analoges gilt für Δ_{A_O} und ρ_{A_O}.

Die Meta-Planung β^1 definieren wir daher für jede Transformation

$$\lambda : (N_1, \mathcal{O}_1, R_1, D_1, \psi_1, Ag_1) \mapsto (N_2, \mathcal{O}_2, R_2, D_2, \psi_2, Ag_2)$$

als die folgende Abbildung, die wir auch als mit $\beta^1(\lambda)$ bezeichnen:

$$\begin{array}{rcl}
\rho_{A_O}(D_0) & \mapsto & \{R_2(t^\bullet) \mid t \in Ag_2^{-1}(A_O) \cap T_2, D_2(t) = D_0\} \\
\tau^3_{A_O}(R) & \mapsto & \beta^1(\tau^3) \text{ mit } \beta^1(\tau^3)(R) \in P_{2,RD}(R) \\
\Delta_{A_O}(G)(b) & \mapsto & \{ \quad (D_2(t), \{(R_2(p), A_p) \mid p \in t^\bullet\}) \\
 & & \mid \quad t \in (T_2 \cap \phi(b)^\bullet \cap Ag_2^{-1}(A_O)), \forall p \in t^\bullet : A_p \in \mathcal{A}_{dlg}(p)\}
\end{array}$$

Welches Bild wir hierbei für $\beta^1(\tau^3)(R)$ definieren, ist gleichgültig, solange es nur in $P_{2,RD}(R)$ liegt.

Bislang hatten wir beliebige Modifikation β^1 als Meta-Planung zugelassen. Mit den Transformationsteams TT einer transparenten Organisation haben wir nun ein Kriterium, um die Wohlgeformtheit einer Meta-Planung zu charakterisieren.

Definition 58 *Eine Meta-Planung β^1 eines Positionsagenten O ist* wohlgeformt*, wenn $\lambda(TT)$ eine wohlgeformte Transformation ist und β^1 durch das Transformationsteam TT induziert wurde:*

$$\beta^1 = \beta^1\left(\lambda(TT)|_O\right) \quad \textit{mit} \quad \lambda(TT)|_O := \bigoplus_{e \in (E \cap \phi^{-1}(O))} \lambda(\phi(e))$$

Hierbei bezeichnet $\lambda(TT)|_O$ den Anteil der Transformation, der O betrifft.

Man beachte, dass $\bigoplus_{e \in (E \cap \phi^{-1}(O))} \lambda(\phi(e))$ definiert ist, da nach Definition alle Transformation einer Position zueinander nebenläufig sind und in einem Teamnetz keine Transition t durch zwei Ereignisse abgebildet werden kann.

Wir wissen also, dass jeder Prozess einer wohlgeformten, transformierenden Organisation eine lokale Transformation darstellt und sich nach Theorem 50 alle Änderungen durch *lokale Modifikation an den Positionsagenten* vornehmen lassen. Die Kommutativität

$$\mu \circ f = g \circ \mu$$

des Diagramm in Abbildung 5.2 wird also erreicht, wenn wir $f = \lambda(TT)$ und $g = \beta^1(\lambda(TT)|_O)$ setzen:

$$\mu\big((\lambda(TT)(Org_1, Ag_1)\big) = \beta^1(\lambda(TT)|_O)\left(\mu(Org_1, Ag_1)\right)$$

5.4 Elementare Tranformationen

Es gibt unendlich viele Transformationen, da es unendlich viele MAS-Organisationen gibt. Um eine unendlich Menge von Transformationen endlich zu beschreiben, gehen wir davon aus, dass es eine endliche Menge an Transformationsmustern gibt, die diese Klasse von Transformationen definiert. Eine Organisationstransformation bildet Netze, also spezielle Graphen, mit Anschriften aufeinander ab. Wir können also *Graphersetzungssysteme* verwenden, um die Transformationen darzustellen (vgl. Padberg u. a., 1995, 1998; Hoffmann u. a., 2005). Graphersetzungssysteme verallgemeinern Termersetzungssysteme in dem Sinne, als dass Termersetzungssysteme nur auf speziellen Graphen, nämlich den Termbäumen operieren, während Graphersetzungssysteme beliebige Graphen betrachten. Die Regeln eines Graphersetzungssystems erlauben es, einen Teilgraphen in einem beliebigen Kontext zu substituieren. Da der Kontext beliebig ist, kann eine solche Regel auf unendlich viele Instanzen angewendet werden.

Die Regeln eines Graphersetzungssystems bilden somit eine Menge der gesuchten Muster. Da jede Regel eine Teilgraphen in einem beliebigen Kontext substituieren kann, ist sie auf unendlich viele Instanzen anwendbar. Dabei ist es nicht notwendigerweise so, dass jede beliebige Regelmenge die Menge aller Transformationen generiert. Im allgemeinen generiert sie nur eine Teilmenge aller möglichen Transformationen. Dies ist typischerweise auch gewollt, denn Organisationstransformationen können unterschiedlich schwer zu implementieren sein, unterschiedliche Kosten verursachen usw., so dass man sich auf solche Transformationen einschränkt, die gute Eigenschaften besitzen.

Wir studieren nun einige Transformationsmuster, die entweder naheliegend sind oder die geeignet sind, wohlgeformte Organisationen in solche zu überführen. Die Anpassungen beziehen sich dabei sowohl auf das Organisationsnetz, d.h. wir nehmen Transformationen auf der Graphenstruktur des Netzes vor, als auch auf die Anschriften R, D und ψ sowie die Organisationsstruktur $\mathcal{O}$.

Vergröberung und Verfeinerung von Positionen Bereits in Definition 47 haben wir das Konzept der Vergröberung für Positionsmengen definiert. Jede Vergröberung ist eine Transformation, ebenso wie die dazu inverse Transformation, die Verfeinerung. Diese Transformation ist in Abbildung 5.5 skizziert.

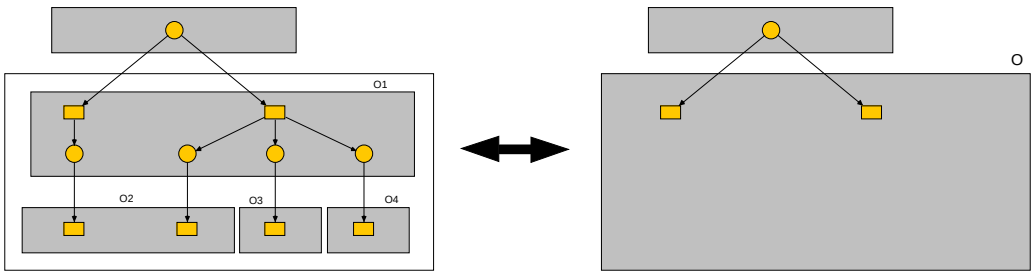


Abbildung 5.5: Vergröberung der Position

Vergröberung und Verfeinerung von Tasks Eine Verfeinerung einer Tätigkeit t kann sinnvoll sein, falls die Menge der zugelassenen Prozesse $\psi(t)$ in zwei Teile zerfällt: $\psi(t) = \psi_1 \vee \psi_2$ mit $D(t_1) = D(t_2) = D(t)$. In diesem Fall bestehen eigentlich

zwei Aufgabenbereiche, die durch die zwei Tätigkeiten t_1 und t_2 mit $\psi(t_1) = \psi_1$ und $\psi(t_2) = \psi_2$ realisiert werden können. Umgekehrt können so zwei Transitionen mit gleichem Dienstnetz und gleichem Nachbereich zu einer Transition zusammengefasst werden. Dies ergibt die inverse Transformation. Diese Transformation ist in Abbildung 5.6 skizziert.

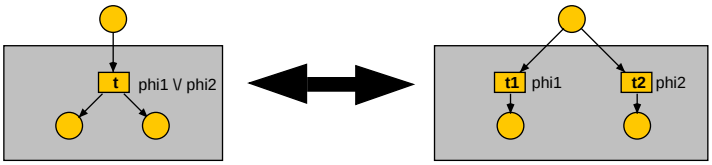


Abbildung 5.6: Transformation der Steuerung

Aufspaltung und Verschmelzung von Positionen Eine Position O, die aus zwei Mengen besteht: $O = O_1 \cup O_2$ mit $O_1 \cap O_2 = \emptyset$, kann auf die beiden Positionen aufgeteilt werden, wenn sie sich gegenseitig nicht nutzen d.h. wenn gilt:

$$(T \cap O_i)^\bullet \subseteq O_i \quad \text{für } i = 1, 2$$

Diese Transformation ist auch invers sinnvoll. Sie beschreibt die Verschmelzung zweier Positionen. Diese Transformation ist in Abbildung 5.7 skizziert.

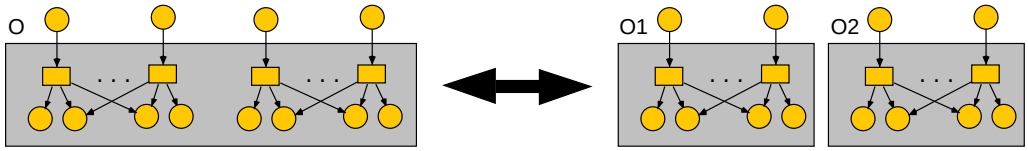


Abbildung 5.7: Transformation der Position

Umlenken von Delegationsbeziehungen Wenn wir eine Delegationsbeziehung zwischen O_1 und O_2, d.h. eine Kante (p_1, t_2) modifizieren wollen, dann ist es für den Erhalt der Wohlgeformtheit wichtig, dass die Aufgabe p_1 weiterhin bearbeitbar ist, d.h. es muss mindestens eine weitere Transition $t \in p_1{}^\bullet$ geben, die die Aufgabe von t_2 übernehmen kann. Außerdem darf durch das Entfernen der Kante (p_1, t_2) die Transition nicht isoliert werden, d.h. wir müssen ${}^\bullet t \neq \emptyset$ sicherstellen, d.h. es muss eine weitere Stelle p existieren, deren Aufgabe durch t_2 bearbeitet werden kann. Zu dieser fügen wir dann die Kante (p, t_2) ein. Die Delegationsbeziehung (p_1, t_2) wird so zur Relation (p, t_2). Die Beziehung wird also nur umgelenkt. Diese Transformation ist in Abbildung 5.8 skizziert.

Generieren und Löschen von Delegationsbeziehungen Eine Alternative zum Umlenken von Delegationsbeziehung zwischen O_1 und O_2 ist das Löschen, bzw. invers dazu: das Generieren. Soll eine Kante (p_1, t_2) gelöscht werden, dann ist es weiterhin notwendig, dass die Aufgabe p_1 weiterhin durch mindestens eine weitere Transition $t \in p_1{}^\bullet$ bearbeitbar bleibt. Nach dem Entfernen der Kante besitzt die Transition t_2 einen leeren Vorbereich. Besitzt sie auch einen leeren Nachbereich, so können wir sie direkt löschen. Ist dies nicht der Fall, so ist darauf zu achten, dass für

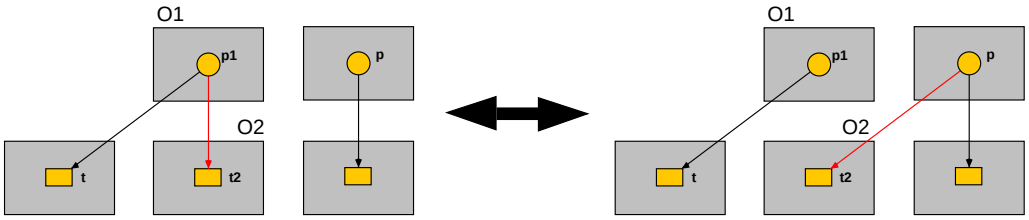


Abbildung 5.8: Transformation der Delegationsbeziehungen

jede Stelle $p \in t_2^\bullet$ eine alternative Transition $t_p \in {}^\bullet p$ existiert, damit keine Stellen mit leerem Vorbereich existieren. Stellen mit leerem Vorbereich sind zwar nicht verboten, aber wir wollen auch noch die Initialitätseigenschaft (3.6) erfüllen, d.h. für alle Stellen $p \in P_N$ mit ${}^\bullet p = \emptyset$ gilt $\forall t \in p^\bullet : R(p) = R(D(t))$. Diese Transformation ist in Abbildung 5.9 skizziert.

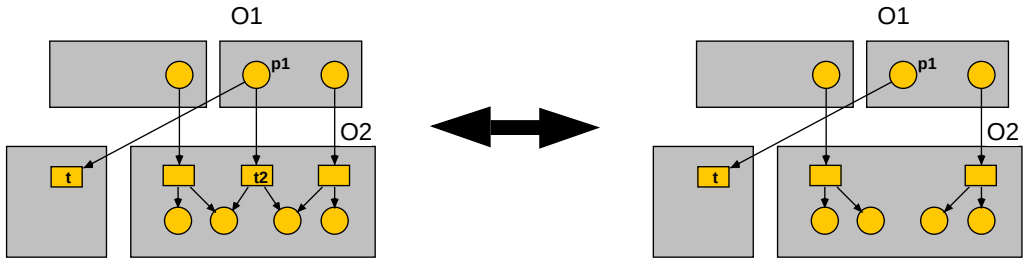


Abbildung 5.9: Löschen/Einfügen der Delegationsbeziehungen

Generieren/Löschen von Tätigkeiten und Aufgabenprofilen Eine weitere Transformation besteht darin, neue Tätigkeiten und Aufgabenprofile zu generieren. Wir können in die Position O eine neue Tätigkeit t hinzufügen, wenn es bereits eine Stelle p gibt, deren Rollenprofil t realisieren kann. Ist die Transition t ihrerseits delegierend, d.h. $t^\bullet \neq \emptyset$, dann müssen alle $p_i \in t^\bullet$ mit bereits bestehenden Transitionen verknüpft werden, damit die Wohlgeformtheit gewahrt bleiben kann. Darüberhinaus sind natürlich noch die Bedingungen an die Rollen- und Verhaltensverfeinerung zu beachten. Diese Transformation ist in Abbildung 5.10 skizziert.

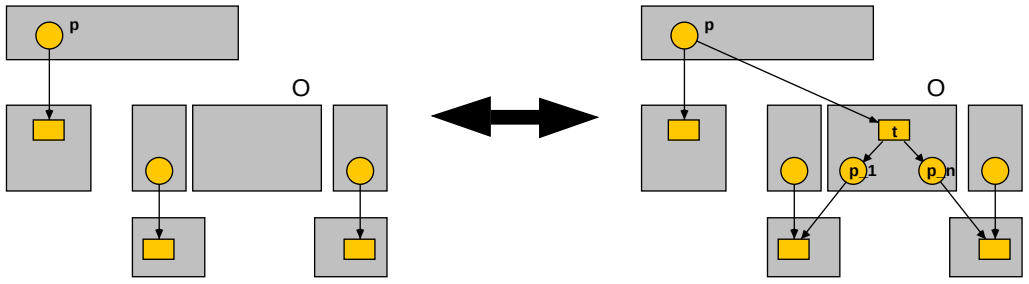


Abbildung 5.10: Löschen/Einfügen von Tätigkeiten und Aufgabenprofilen

5.5 Organisationsmetriken

Um eine Meta-Planung geignet operationalisieren zu können, müssen wir Optimalitätsfaktoren definieren, die bewerten, ob eine Transformation die Struktur verbessert oder verschlechtert. Dazu definieren wir ein Maße, die eine Organisation bewerten. Diese Maße sind Einflussparameter für die Transformation, denn im allgemeinen lohnt es sich nur dann, eine Organisation zu transformieren, wenn sich die Bewertung verbessert.

Nehmen wir an, dass es n verschiedene Bewertungsmaße einer Organisation gibt und dass sich diese reellwertig quantifizieren lassen. Eine *Organisationsmetrik* ist dann eine Abbildung:

$$v : \mathcal{ORG} \to \mathbb{R}^n$$

Definiere für die Transformationen $f : (Org_1, Ag_1) \mapsto (Org_2, Ag_2)$ die Werte $v_1 = v(Org_1, Ag_1)$ und $v_2 = v(Org_2, Ag_2) = v(f(Org, Ag))$. Dann gibt es verbessernde, verschlechternde und unvergleichbare Transformationen f, nämlich dann, wenn mit der partiellen Ordnung $\leq$ auf Vektoren gilt:

Verbesserung:	$v_1 \leq v_2$
Verschlechterung:	$v_1 \geq v_2$
Unvergleichbarkeit:	$v_1 \not\leq v_2 \wedge v_1 \not\geq v_2$

Der Fall der Unvergleichbarkeit tritt also dann auf, wenn f die Bewertung bezüglich einiger Kriterien verbessert, gleichzeitig aber bezüglich einiger anderer Kriterien verschlechtert. Eine eindeutige Bewertung von f ist dann also nicht möglich.

Jede Transformation benötigt Ressourcen, beispielsweise Zeit oder Geld. Wir nehmen an, dass es m verschiedene Ressourcentypen gibt. Wir treffen die Annahme, dass sich die Transformation f in ihrem Bedarf für alle Argumente (Org_1, Ag_1), d.h. für alle Organisationen, gleich verhält. Diese Annahme der Uniformität ist plausibel, da unsere Transformationen sich von Transformationsmustern ableiten, und diese verhalten sich für alle Organisationen gleich, da sie nur auf ihrem Muster arbeiten und den weiteren Kontext, in dem sie das Muster vorgefunden haben, ausblenden. Unter dieser Annahme lassen die Kosten einer Transformation durch eine Abbildung in die nicht negativen Zahlen modellieren:

$$c : \mathcal{TM} \to \mathbb{R}^n_+$$

Für eine gegebene Transformation f, die wir auf eine MAS-Organisation (Org, Ag) anwenden, berücksichtigen wir die drei Größen

$$v_1 = v(Org, Ag), \qquad v_2 = v(f(Org, Ag)) \qquad \text{und} \quad c(f),$$

um zu entscheiden, ob die Transformation f vorgenommen werden soll. Hierbei können die drei Größen unterschiedlich verknüpt werden. Ein einfaches Kriterium ist es, im Falle $v_1 \leq v_2$ den Quotienten aus der absoluten Verbesserung $v_2 - v_1$ und den Kosten heranzuziehen, also: Verbesserung pro Kosten. Die Verbesserung gewichten wir dabei noch linear mit $a \in \mathbb{R}^n_+$:

$$eval(f, Org, Ag) := \frac{a(v(f(Org, Ag)) - v(Org, Ag))}{c(f)}$$

Der Gewichtungsfaktor a kann im Falle unvergleichbarer MAS-Organisationen entscheiden, welche der n Kriterien stärker zu berücksichtigen sind.

Mit diesem Maß versehen ist es naheliegend, unsere Reorganisation so zu gestalten, dass $eval(f, Org, Ag)$ maximiert wird. Reorganisation reduziert sich damit dann zu einem Optimierungsproblem. Offen ist bislang noch, auf welche Maße sich die Metrik stützt. Betrachten wir im folgenden einige naheliegende Kandidaten.

Zunächst halten wir fest, dass wir bereits ein grundlegendes Instrument zur Bewertung der Organisationsstruktur kennengelernt haben, nämlich die Wohlgeformtheit und die Charakterisierung der Tätigkeitspfade bezüglich der sicheren Bearbeitbarkeit von Aufgaben. Dieses Maß ist ein ausschließendes, da nur wir nur wohlgeformte Organisationen heranziehen wollen. Betrachten wir nun, welche weiteren möglichen Bewertungen existieren.

Organisationsnetz Die Wohlgeformtheit einer Organisation impliziert bereits eine Vielzahl wünschenswerter Eigenschaften. So gibt es keine unnützen Teams, es gibt keine Teamprozesse, die nicht korrekt gesteuert würden usw. Die Organisation ist somit funktional korrekt – ein für den praktischen Einsatz wichtige Eigenschaft, die wir hier aber nicht in die Bewertung mi taufnehmen können, da wir ja nicht wohlgeformte Organisationen von der Betrachtung ausgeschlossen haben.

Darüberhinaus können wir die Organisation als Netzwerk analysieren. Eine Kenngröße ist der Alternativgrad einer Organisation, die sich in der Anzahl der Alternativen bemisst, die zu einer Ausführung einer Aufgabe herangezogen werden können. Diese Größe können wir folgendermaßen quatifizieren:

$$\max\{|p^\bullet| : p \in P\} \tag{5.3}$$

Eine damit verwandte Kenngröße misst die Delegationsverzweigung, d.h. die Anzahl der Subtasks, die zur Bewältigung einer Aufgabe herangezogen werden. Diese Größe können wir folgendermaßen quantifizieren:

$$\max\{|t^\bullet| : t \in T\} \tag{5.4}$$

Organigramme Faltet man alle Stellen und alle Transitionen einer Position $O \in \mathcal{O}$ zu einem Knoten, so erhält man einen Graphen, der das Organigramm der abstrakten Organisation darstellt. Dieser Graph ist im allgemeine nicht zyklenfrei, wie man anhand des Organisationsnetzes in Abbildung 5.11 und seiner Vergröberung in Abbildung 5.12 erkennen kann.

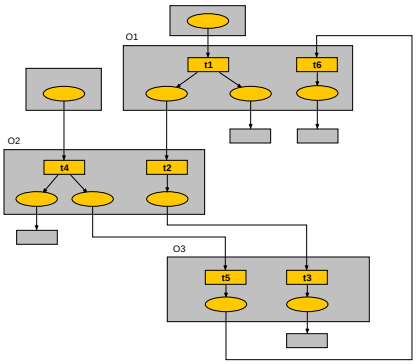


Abbildung 5.11: Organisation

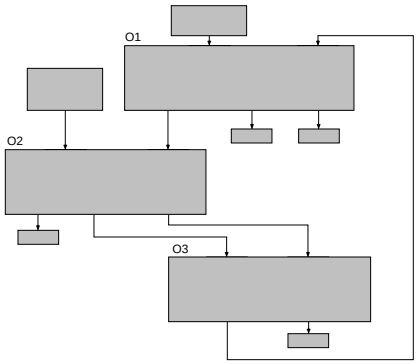


Abbildung 5.12: Vergröberung

Durch die Vergröberung werden durch die Kanten nur noch abstrakte Kommunikationsbeziehungen beschrieben, bei denen die jeweilige Tätigkeit und deren Ab-

hängigkeiten ausgeblendet werden. Es ist offen, ob die Kommunikationsbeziehungen statisch, d.h. für alle Aktivitäten gleich sind, oder ob sie vom Aufgabenprofil abhängen.

In dem Organisationsnetz in Abbildung 5.11 existiert der Abhängigkeitspfad $t_1t_2t_3$, der eine Hierarchie der Positionen von O_1 über O_2 zu O_3 nahelegt. Diese Hierarchie deckt sich jedoch nicht mit der des Abhängigkeitspfade $t_4t_5t_6$ der von O_2 über O_3 zu O_1 verläuft. In betrieblichen Organisationsstrukturen ist es vorteilshaft, die Organisationsstrukturen (insbesondere die Weisungsstrukturen) für alle Aufgabenpfade identisch zu gestalten, um so einen aufwändigen Kontextwechsel der Delegationsbefugnisse zu vermeiden.

Die Zyklenfreiheit des Graphen erlaubt es, auf die Zyklenfreiheit der Tätigkeitspfade, bzw. ihre Endlichkeit zu schließen. Die Umkehrung gilt jedoch, wie wir gesehen haben, im allgemeinen jedoch nicht. Generell ist eine Modellierung einer Organisation, deren Organigramm bereits zyklenfrei sein muss, zu eingeschränkt. In diesem Fall ist es notwendig, die Tätigkeitspfade selbst zu analysieren. Ein Organisationsmaß sollte daher Organisationsnetze besser bewerten, wenn ihr Organigramm zyklenfrei ist.

Organigramme leisten darüberhinaus auch beim Entwurf des Multiagentensystems einen wichtigen Beitrag, denn wenn wir zum Organigramm einen aufspannenden Baum konstruieren, dann erhalten wir einen hierarchischen Namensraum, der geeignet ist, dass System auf verschiedene physikalische Rechner zu verteilen.

Rollenredundanz Rollen- und Dienstnetzzuweisung liefern wichtige Bewertungen beim Zuschnitt der Position. Die Positionen sollen nämlich einerseits allumfassend sein, um die Kommunikationskosten und den Overhead, der durch Aufteilung der Aufgaben entsteht, gering zu halten. Andererseits ist eine Verteilung und Spezialisierung wünschenswert, da sie eine leichte Wartbarkeit und Lastverteilung ermöglichen.

Die Rollenredundanz bezogen auf Positionen können wir berechnen, indem wir $R({}^\bullet O \cap T)$ als Multimengen betrachten und damit die folgende Multimenge definieren:

$$Red(N) := \sum_{O \in \mathcal{O}} R({}^\bullet O \cap T)$$

Für jede Rolle $r \in Rol$ gibt $Red(N)(r)$ die *Redundanz* von r an. Das Rollenredundanzmaß ergibt sich dann folgendermaßen:

$$\frac{\sum_{r \in Rol} Red(N)(r)}{|Rol|} \tag{5.5}$$

Die Redundanz schätzt grob die Spezialisierung der einzelnen Positionen ab. Eine hohe Spezialisierung ist einerseits sinnvoll, da sie die Teamformation erleichtert, andererseits ist Redundanz sinnvoll, um Alternativen zu besitzen.

Dienstüberlappung Wir wollen nun betrachten, wie sehr sich die Positionen bezüglich der von ihnen angebotenen Dienste $D(T \cap O)$ überlappen. Dazu müssen wir definieren, wann sich Dienste überschneiden. Die in einem Dienstnetz verwendeten Typen liefern einen Hinweis, wie groß die Übereinstimmung ist, vorausgesetzt natürlich, dass das Typsystem, d.h. die Systemenontologie, fein genug modelliert ist. Die Menge der in D verwendeten Typen ist:

$$\{d(p) \mid p \in P_D\}$$

Zwei Dienste D_1 und D_2 stehen miteinander in Verbindung, wenn es einen Typ gibt, der in beiden genutzt wird. Diese Relation auf Dienstnetzen bezeichnen wir mit R_{Dienst}:

$$(D_1, D_2) \in R_{Dienst} :\Longleftrightarrow \exists p_1 \in P_1 : \exists p_2 \in P_2 : d(p_1) = d(p_2)$$

Man kann die Aussagekraft noch verbessern, wenn man elementare Typen wie Integer usw. von der Betrachtung ausnimmt. Die Relation R_{Dienst} ist reflexiv und symmetrisch. Wir können daher die Bezirke $C \in \textsc{Bez}(R_{Dienst})$ der Relation R_{Dienst} berechnen, d.h. die maximalen Mengen von Diensten, die paarweise miteinander in Relation stehen. Die Bezirke entsprechen in der ersten Näherung Datenspeichern, die verwandte Datentypen und ihre Instanzen aufnehmen. Typen, die nicht gemeinsam verwendet werden, können in verschiedenen Datenspeichern abgelegt werden.

Das Überlappungmaß ist die kumulierte Anzahl der Gemeinsamkeit der Positionen mit den Bezirken:

$$\sum_{O \in \mathcal{O}} |\{C \in \textsc{Bez}(R_{Dienst}) : C \cap D(T \cap O) \neq \emptyset\}| \tag{5.6}$$

Dieser Wert ist besonders groß, wenn Positionen an vielen Bezirken partizipieren, d.h. wenn sie an vielen Datenspeichern angeschlossen sind.

Diese Information können wir ebenfalls für die Implementation nutzen, denn die von mehreren Positionen gemeinsam genutzten Typen definieren Cluster, die zum schnelleren Zugriff auf die mit den Typen assoziierten Daten auf einem physikalischen System verwaltet werden. Diese Anforderung tendiert dazu, zentralisierte Ansätze zu bevorzugen. Dies steht meist jedoch im Widerspruch zu der Forderung nach einer physikalischen Dezentralisierung. Zwischen diesen beiden Punkten muss ein Ausgleich gefunden werden. Hierbei ist das Überlappungsmaß ein wichtiges Hilfsmittel.

Teamwork Eine weitere interessante Größe betrifft die Teamaktivität. Hierbei versuchen wir abzuschätzen, wie positiv sich die organisationale Koordinierung durch die Teambildung und die Steuerung auswirkt. Es ist anzustreben, dass die Organisation für den einzelnen Agenten eine Reduktion des Koordinierungsoverheads bedeutet, und dass die Steuerung eine Vergrößerung des Entscheidungsspielraum bedeutet, da „irrelevante“ Möglichkeiten ausgeblendet werden. Die Gefahr, die dabei droht, ist natürlich, auch relevante, interessante Prozesse auszublenden. Wollen wir dies quantifizieren, so sind wir darauf angewiesen, dass wie jedem Ablauf eines Dienstes eine Bewertung geben können. Sei $M_0 \xrightarrow{w} M_f$ ein korrekter Ablauf von D. Diesen bewerten wir mit der Abbildung v_π.

Nun vergleichen wir zu einer Rolle $R \in \mathcal{R}$ den Mittelwert $\bar{v}_0$ aller Abläufe der Menge

$$\bar{v}_0 := \text{avg}\left\{ v_\pi(w) : M_0 \xrightarrow[D]{w} M_f, R(D) = R, D \in \mathcal{D} \right\}$$

mit dem Mittelwert $\bar{v}_{Org}$ der Menge aller Abläufe, den die gesteuerten Teams $D(G) \times \hat{\psi}(G)$ der Organisation erzeugen:

$$\bar{v}_{Org} := \text{avg}\Big\{ v_\pi(w) : M_0 \xrightarrow[D(G)\times\hat{\psi}(G)]{w} M_f, \quad G = G(K, \phi), \\ (K, \phi) \in \mathcal{K}^{pg}(N, \{p\}), p \in P_{Org}(R) \Big\}$$

Der Quotient der beiden Größen ist dann ein Maß für die Verbesserungen durch die organisationelle Koordinierung:

$$\frac{\bar{v}_{Org}}{\bar{v}_0} \tag{5.7}$$

Natürlich hängt das Maß in problematischer Art und Weise von der geschickten Wahl der konkreten Bewertung der Abläufe $v_\pi(w)$ ab. Sie generell zu definieren ist praktisch nicht möglich. Nur wenn man den einzelnen Tätigkeiten t von D Kosten und Nutzen zuweisen kann, mag dies für diesem konkreten Spezialfall gelingen.

Daneben kommt es auch auf die Größe des Regelwerks an, denn die Regeln müssen leicht zu befolgen sein, damit sie die Agenten internalisieren können. Außerdem stellt sich die Frage, ob überhaupt noch die richtigen Prozesse geregelt werden, oder ob man mit einer Vielzahl an Regeln versucht, solche Aktivitäten auszuschließen, die von den Organisationsmitgliedern aufgrund externer Einflüsse gar nicht gewählt werden.

Zusammenfassung

In diesem Kapitel haben wir Reorganisationsprozesse als spezielle organisationale Aktivität behandelt. Reorganisation werden durch Organisationstransformationen vorgenommen, d.h. durch formale Abbildungen, die Organisationen wiederum auf Organisationen abbilden.

Wir haben speziell solche Transformationen untersucht, die vom System selbst erbracht werden. Diese Selbstorganisation wird durch Transformationsteams erbracht, in denen jeder Agent nur spezielle Transformationen vornehmen darf, nämlich eine Teilmenge der lokalen Transformationen, die wir als transparent bezeichnet haben, weil ihre Wirkung auf die Interna des transformierenden Agenten beschränkt bleibt. Aus diesem Grund lässt sich die auf den Agenten lokalisierte Transformation als Meta-Planung des SONAR-Agenten auffassen, da diese ja ebenfalls die Interna modifiziert.

Weiterhin haben wir untersucht, welche elementaren Tranformationen auf Organisationen wir unterstützen wollen, nämlich beispielsweise die Vergröberung und Verfeinerung von Positionen oder von von Tasks; die Aufspaltung und Verschmelzung von Positionen; das Umlenken von Delegationsbeziehungen; das Generieren und Löschen von Delegationsbeziehungen oder von Tätigkeiten und Aufgabenprofilen. Dies erzeugt einen Basissatz an Transformation.

Für die Agenten ist darüberhinaus noch wichtig, entscheiden zu können, wann Transformationen nützlich sind. Dazu muss der Agent den Aufwand einer Transformation zu der Verbesserung ins Verhältnis setzen. Um die Verbesserung einer Transformation bewerten zu können, haben wir Organisationsmetriken eingeführt, die jeder Organisation eine mehrdimensionale Bewertung auf Basis des Organisationsnetzes, des Organigramms, der Rollenredundanz und der Dienstüberlappung zuweist.

Nach dieser Untersuchung der formalen Organisation widmen wir uns im folgenden Kapitel den Wechselwirkungsformen mit den informellen Anteilen.

6 Agentensysteme als Interaktion von Organisation und Mitgliedern

Eine Organisation ergibt sich als die Verschränkung der formalen Organisation mit ihrem informellen Anteil, der sich durch die Organisationsmitglieder ergibt. In unserem Fall geschieht dies, indem wir jeden Positionsagenten mit einem Mitgliedsagenten koppeln. Der Mitgliedsagent füllt den Positionsagenten mit „Leben“ an.

6.1 Positionen und Organisationsmitglieder

Betrachte die komplette Organisation in Abbildung 6.1, bei der sich zwei Firmen zu einer virtuelle Organisation zusammengeschlossen haben. Die Elemente, die sich in der Abbildung innerhalb des mit „virtuelle Organisation“ beschrifteten Bereichs befinden, stellen die formale Organisation dar, die aus der Organisationsstruktur, den Positionen (hier: der Initiator, der Koordinator und die beiden Firmenrepräsentanten) und den Aufgaben besteht. In der agentenorientierten Umsetzung des Modells wird jede Position durch einen Agenten implementiert. Obwohl jetzt Agenten im Spiel sind, handelt es sich immer noch um die formale Organisation, denn die Agenten implementieren nur die mit der Position verbundenen Handlungsstrukturen. Von der Position streng zu trennen ist aber der Positionsinhaber – das Organisationsmitglied.

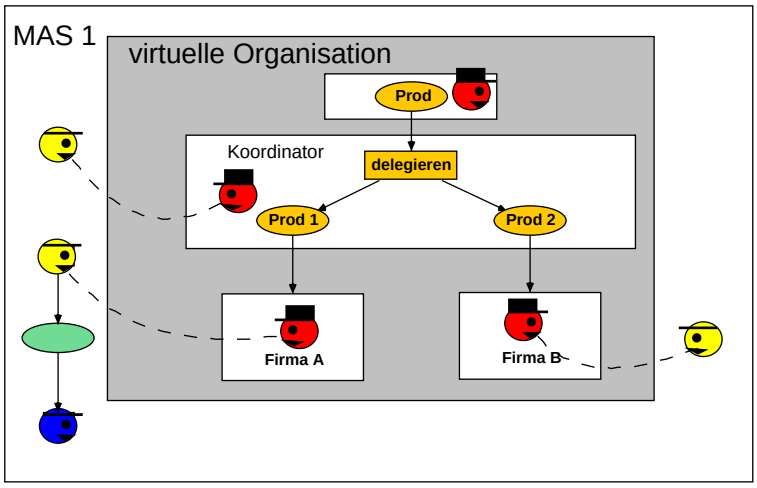


Abbildung 6.1: Implementation der formalen Organisation

Um die Organisation zu erhalten, müssen wir alle Positionen mit Mitgliedern besetzen. In der Abbildung ist dies durch zusätzliche eingefügte Agenten dargestellt. Man beachte, dass diese *Mitgliedsagenten* für die Organisation *extern* sind, denn Organisationsmitglieder sind nach strenger theoretischer Auffassung nicht Bestandteil der Organisation, sondern nur durch Mitgliedschaft mit ihr assoziiert – in

der Abbildung durch eine gestrichelte Linie dargestellt. Der Mitgliedsagent, der die Position implementiert, kommuniziert mit dem Positionsagenten, der seine einzige Schnittstelle zur Organisation darstellt. Durch die Rollen der Position ist festgelegt, welche Fähigkeiten der Agenten mitzubringen hat und welche Aktivitäten er ausführen muss. Formale Organisation und ihre Mitglieder bilden zusammen die informelle Organisation. Daneben gibt es auch noch die Agenten, die mit keiner Positionen assoziiert sind. In der Abbildung 6.1 ist diese der Agent links unten. Solche Agenten sind weder Bestandteil der formalen noch der informellen Organisation.

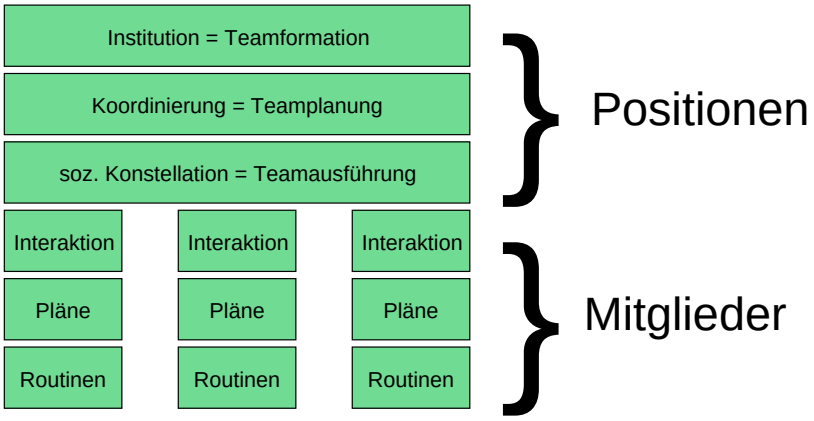


Abbildung 6.2: Dualität von Positionen und Mitgliedern

Die Position ist ein Akteur, der mit Abläufen auf die Umwelt reagiert. Diese Abläufe werden zwar von Mitgliedern der Organisation getragen, sind jedoch gleichzeitig in ihrer Existenz von ihnen unabhängig, da ihre Grundstruktur von den Positionsagenten, also der formalen Organisation, bereits festgelegt ist. Es liegt nahe, das in Kapitel **??** dargestellte Akteursmodell unter dem Blickwinkel von formalen wir informalen Organisationselementen zu betrachten, denn die Art und Weise, wie die Position agiert, entspricht denen eines SONAR-Agenten (vgl. dazu Abbildung 6.2). Die von den Positionen getragenen organisationalen Transformationen stellen die Institutionalisierungsprozesse dar, die Teamplanung entspricht der Koordination und die Teamformation erzeugt die institutionalisierten Konstellationsarten, innerhalb dessen die Mitglieder agieren. Wir erkennen hier die in Abbildung **??** dargestellte selbstähnliche Korrespondenz von Mikro- und Makroprozessen wieder. Positionen und Mitglieder sind somit unterschiedliche Perspektiven auf das Organisationsgeschehen, die beide geleichermaßen relevant sind und auch beide in der SONAR-Architektur repräsentiert werden.

Diese Trennung in Positions- und Mitgliedsagenten mag einem auf den ersten Blick redundant erscheinen. Sie ist aber auch in hohem Maße auch im praktischen Kontext der Software-Entwicklung nützlich, nämlich immer dann, wenn die Entwicklung der Organisation getrennt von der ihrer Mitglieder vorgenommen wird. Dies ist beispielsweise typisch für die Entwicklung von Handelsplattformen aus dem Bereich des *e-commerce*, hier speziell der Bereich: *business to business (B2B)*. Hier sollen Agenten verschiedener Unternehmen bei der Anbahnung und Abwicklung elektronischer Verträge unterstützt werden. Aus Sicht der Plattform sind dazu Handelsregeln für die Preisfindung und die Vertragsabwicklung zu entwickeln. Da die Agenten der Marktteilnehmer jedoch in den Unternehmen entwickelt werden,

kann die Entwicklung nur die formale Organisation betrachten. Bei der Analyse der Plattform können daher keine spezifische Eigenschaften der Mitgliedsagenten angenommen werden. Daher ist allein eine Kontrolle der Mitglieder durch die von den Positionsagenten überwachten organisationalen Handelsregeln in diesem Szenario sinnvoll. Von dieser Trennung unbenommen bleibt die Tatsache, dass das Wechselspiel von konkreten Positionen und Mitgliedern eine adjustierende Anpassung der Handelsregeln – im Sinne einer Ordnungspolitik der Organisation – nötig werden lässt.

Man beachte, dass die Organisation nur bestimmt, wie sich die Rollen und Positionen der Organisation zueinander verhalten. Es wird nicht erzwungen, dass ihre Mitglieder mit den Positionen identisch sind. Das implementierende Multiagentensystem besteht daher mindestens aus den Mitglieder, der informellen Organisation also, kann aber auch weitere externe Agenten enthalten, mit denen die Mitglieder informelle Kanäle unterhalten (vgl. Abbildung 6.1).

Bei der Besetzung besteht nicht notwendigerweise eine direkte Korrespondenz von Position und Mitglied. Es ist auch möglich, dass eine Agent mehrere Positionen besetzt. So ist es möglich, dass der Repräsentant der Firma A auch gleichzeitig als Koordinator tätig wird.

Der externe Mitgliedsagent kann nur mittels des Positionsagenten an den organisationalen Prozessen teilzunehmen.[1] Positions- und Mitgliedsagent teilen sich die Planung der organisationalen Prozesse. Teamwork umfasst drei Teilaspekte:

1. Die Teamformation: Sie generiert mit Hilfe des Organisationsnetzes und der Teamgenerationsfunktion γ ein Team G.

2. Die Teamplanung: Sie stellt sicher, dass alle lokalen Handlungspläne auch Teampläne sind, d.h. Prozesse, der die Steuerbedingung erfüllen: $\pi_A(G)(D)(p) \in Proc(D(G), \hat{\psi}(G))$.

3. Die Ausführungskontrolle: Die kontrollierte Ausführung des Teamplans stellt zur Laufzeit sicher, dass sich der Mitgliedsagent an das Teamnetz $D(G)$ und an die Teamsteuerung $\hat{\psi}(G)$ hält.

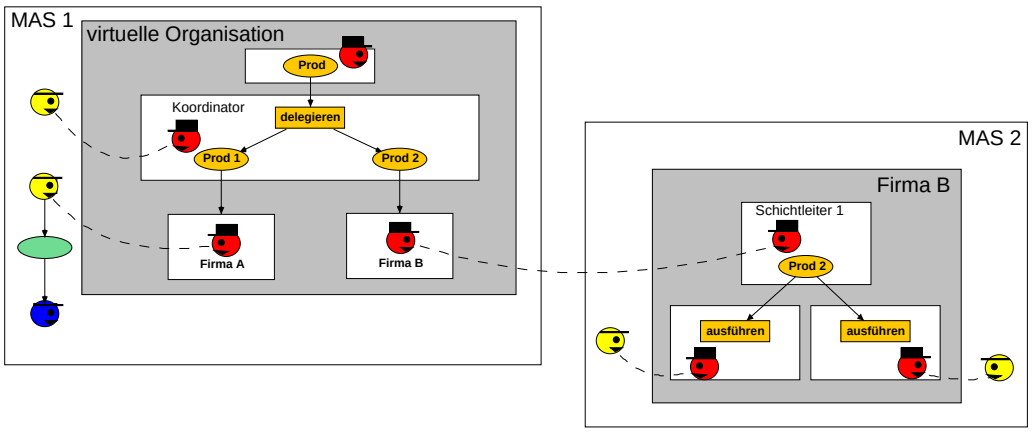


Abbildung 6.3: Organisationen als Organisationsmitglieder

[1] Hier verweisen wir auf die sozialtheoretische Besonderheit, dass es die Rollen der Organisationspostionen sind, die an organisationalen Prozessen teilnehmen, nicht aber – wie man denken könnte – die Akteure, die diese Rollen innehaben.

Diese dualistische Sichtweise passt auch zur Perspektive der Organisationsverbünde. Abbildung 6.3 zeigt den Positionsagenten der Firma B, der nicht durch einen normalen Mitgliedsagenten, sondern wiederum durch eine Organisation implementiert wird. Hier fungieren somit Organisationen als Organisationsmitglieder. Das Modell ist somit rekursiv.

6.2 Die Organisation als Plattform

Aus der Sicht aller Agenten (Positionen, Mitglieder und Externe) dient die Organisation als Dienstleister, gewissermaßen als logische Plattform. Ihre zentrale Aufgabe ist die Verwaltung der Postionsbesetzungen, also die Verbindung von Positionsagenten und Mitgliedsagenten. Dazu hält sie alle offenen Stellenausschreibungen vorrätig, die von allen Agenten eingesehen werden können (vgl. dazu das Protokoll in Abb. 6.4). Im Allgemeinfall wird eine Stelle ausgeschrieben, sobald das aktuelle Mitglied aus der Organisation scheidet. Eine Ausnahme liegt bei der Initialisierung vor, bei der noch gar keine Mitglieder vorhanden sind und daher alle Stellen zunächst ausgeschrieben werden.

Ein Mitglied scheidet aus der Organisation aus, indem das Mitglied den Vertrag kündigt oder ihm von der Position gekündigt wird. In einem realen Multiagentensystem kann dies bedeuten, dass die aktuelle Implementierung der Positionsaufgaben durch eine neuere Version ersetzt wird. Kündigungen modellieren dann den Update-Vorgang in Form einer dynamischen Bindung. Kündigungen können aber auch ein Ausdruck der Fehlerbehandlung sein, die bei der Ausführungskontrolle (s.u.) auftreten kann.

Der betroffene Positionsagent, dessen Mitglied ausscheidet, regelt den Übergang vom ausscheidenden Mitglied zu seinem Nachfolger. Die Mitgliedschaft eines ausscheidenden Agenten endet erst, wenn alle laufenden Teamaktivitäten, in denen das Mitglied involviert ist, beendet sind – und sei es nur, dass sie kontrolliert abgebrochen werden. Der Positionsagent leitet daher die laufenden Teamaktivitäten an den auscheidenden Agenten weiter, während die im Nachfolgenden begonnenen Teamaktivitäten schon vom Nachfolger bearbeitet werden. Im allgemeinen sind also in der Übergangszeit – wenn auch typischerweise nur für relativ kurze Zeit – mehrere Agenten als Besetzung einer Position tätig.

In der praktische Umsetzung wird die Kommunikation von Mitglied und Position sowie innerhalb der Positionen durch kryptographische Techniken abgesichert. Bei der Initialisierung der Organisation werden alle Positionsagenten erzeugt. Außerdem generiert die Organisation für jeden erzeugten Positionsagenten einen asymmetrischen Schlüssel. Bei der Erzeugung wird jedem Positionsagent sein eigener Schlüssel sowie der öffentliche Schlüssel der Organisation mitgeteilt.

Die Organisation verwaltet die öffentlichen Schlüssel aller Positionsagenten und versendet diese – mit seinem eigenem Schlüssel signiert – auf Anfrage. Er agiert also als *Key-Server* der Organisation.

Die Schlüssel der Positionsagenten dienen dazu, die Nachrichten, welche die Positionsagenten einander schicken, zu signieren, so dass deren Herkunft kontrolliert werden kann. Dies verhindert, dass sich externe Agenten als Positionen ausgeben können. In einer Organisation wird also sichergestellt, dass nur Positionsagenten und auch nur solche, die dazu laut Organisationsnetz authorisiert sind, miteinander kommunizieren.

Die Kommunikation zwischen Position und Mitglied wird ebenfalls durch Ver-

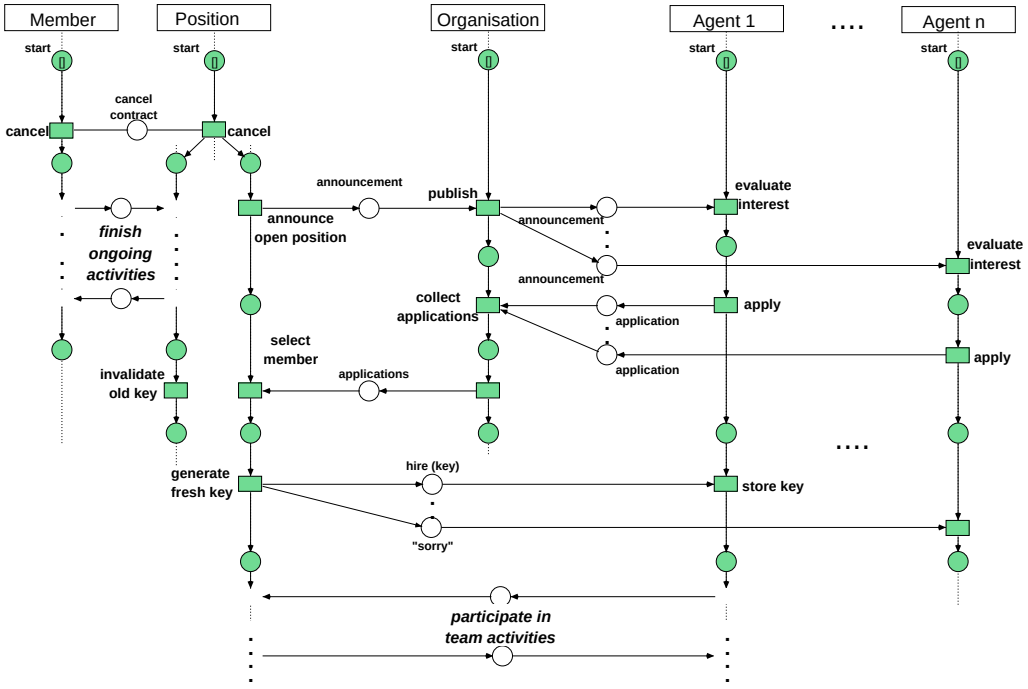


Abbildung 6.4: Protokoll zur Bindung von Position und Mitglied

schlüsselung abgesichert. Dazu generiert der Positionsagent für die aktuelle Besetzung einen frischen Schlüssel und teilt ihn dem Mitgliedsagenten in dem Moment mit, in dem dieser eingestellt wird. Erst die Kenntnis des Schlüssels zeichnet den Agenten als Mitglied der Organisation aus.

Ein Agent kann nur mittels der Position als Organisationsmitglied mit anderen Mitgliedern kommunizieren. Wenn ein Mitgliedsagent eine Nachricht im Kontext seiner organisationalen Tätigkeit versenden will, so schickt er sie zunächst an seinen Positionsagenten. Diese Kommunikation wird durch den gemeinsamen symmetrischen Schlüssel abgesichert. Der Positionsagent signiert die Nachricht und schickt sie an den gewünschten Positionsagenten. Dieser überprüft die Signatur, signiert seinerseits und leitet die Nachricht an sein Mitglied weiter. Durch diesen Mechanismus können die Mitglieder einer Organisation Nachrichten, die ein Agent in seiner Aufgabe als Mitglied versendet, von solchen unterscheiden, die ein Agent als „Privatperson“ versendet.

6.3 Teamwork: Formation, Planung und Ausführung

Die drei organisationalen Aufgaben, die Positionen wahrnehmen, sind – wie wir bereits in Abbildung 4.9 illustriert haben – die Teamformation, die Teamplanung und die Planausführung. Betrachten wir diese Prozesse im folgenden unter Einbezug der Mitglieder.

6.3.1 Teamformation

Die Teamformation geschieht anhand der Teamgenerationsfunktion γ_A. Sie löst – wie bereits dargelegt – den Nichtdeterminismus im Organisationsnetz auf, um ein

Team G zu erzeugen. Jeder Positionsagent A delegiert dabei Teilaufgaben an andere Positionsagenten $A_1, \ldots, A_m$, die ihrerseits rekursiv jeweils ein Teilteam erzeugen, das von A zu einem gemeinsamen Team zusammengesetzt wird.

Hierbei ist essentiell, dass es die Positionsagenten A_i im Prinzip *nicht ablehnen* können, die Teilaufgaben zu bearbeiten. Als Teil Organisation sind sie zur Bearbeitung im Prinzip verpflichtet.

An dieser Stelle formulieren wir die Verpflichtung bewusst mit dem Attribut „im Prinzip“, denn wenn der Agent eine Aufgabe aktuell nicht bearbeiten kann, beispielsweise weil er überarbeitet ist, dann existieren unterschiedliche Szenarien: der Agent bearbeitet die Aufgabe mit großer Verzögerung; der Agent kann eventuell im Ausnahmefall die Aufgabe ablehnen; der Agent kann die Aufgabe ablehnen, muss aber eine Strafe ableisten usw.

Welche der Varianten realisiert wird, hängt von der konkreten Implementation der Interaktion von Position und Mitglied ab. Verschiedene Implementation entsprechen dabei unterschiedlichen Organisationsstrukturen.

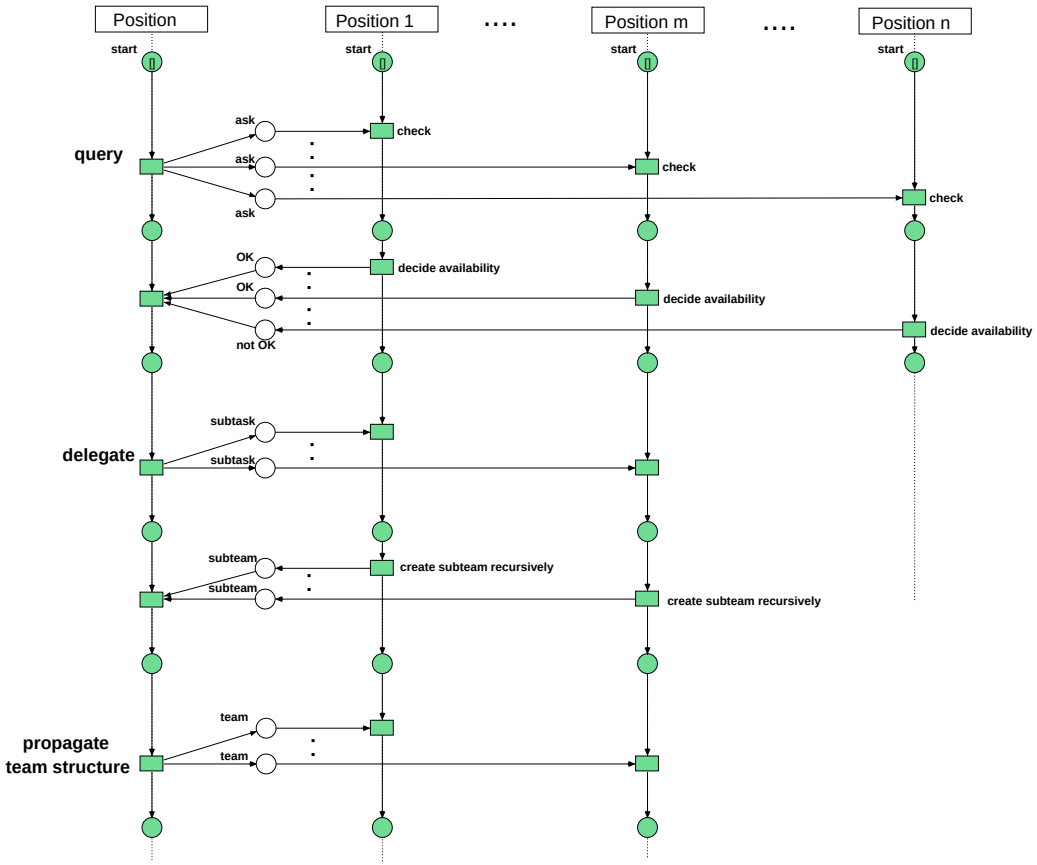


Abbildung 6.5: Protokoll zur Teamformation

Ein Beispielproto koll zur Teamformation zeigt die Abbildung 6.5. Hier fragt der delegierende Agent zunächst bei n Agenten an, ob und wie sie die Aufgabe erfüllen können. Anhand der Antworten entscheidet er sich für eine Teilmenge von m Agenten, an die er delegiert und die die Subteams erzeugen.

6.3.2 Teamplanung

Der Positionsagent gibt durch die Teamplanung, d.h. durch die Generierung von $G = G(K, \phi)$ das Dienstnetze $D = D_\Delta(\delta)[R(p)]$ und die Steuerung $\hat{\psi}(G)$ vor, während der korrespondierende Mitgliedssagent – typischerweise in Absprache mit der Position – den Plan $\pi = \pi(G)(D)(p)$ auswählt:

$$\text{Positionsagent} \quad \underset{\pi}{\overset{D,\hat{\psi}}{\rightleftarrows}} \quad G \quad \text{Mitgliedsagent}$$

In einer konkreten Implementierung wird der Mitgliedsagent eine Präferenz auf allen möglichen Teamplänen $\pi \in \mathit{Proc}(D(G), \hat{\psi}(G))$ besitzen. Die in dem Team G partizipierenden Positionsagenten handeln untereinander aus, wie die Präferenzordnungen der Mitglieder in Einklang gebracht werden, so dass ein Kompromiss in Form eines Teamplans π_G gefunden wird.

Auch hier existieren in Abhängigkeit der Ausprägung der Organisation unterschiedliche Szenarien: der Mitgliedsagent muss den von den Positionsagenten ausgehandelten Kompromiss stets akzeptieren; er kann den Kompromiss ablehnen und nach Anpassung der Präferenz auf eine Neuverhandlung bestehen usw.

Außerdem kann es Autortätsbeziehungen zwischen den Positionen geben, die gegebenenfalls aboslut gelten oder abhängig vom Team sind. Welche der Varianten realisiert wird, hängt auch hier von der konkreten Implementation der Position/Mitglied-Interaktion ab.

Ein Beispielprokoll zur Teamplanung zeigt die Abbildung 6.6, bei dem die Positionsagenten eine Kompromiss aushandeln, wobei der Kompromiss für die Mitgliedern bindend ist.

6.3.3 Planausführung

Da der Positionsagent in wohlgeformten Organisation die Einhaltung der organisationalen Bedingungen garantiert, hat dies zur Auswirkung, dass der Mitgliedsagent eigentlich gezwungen ist, sich gemäß der organisationalen Regeln zu verhalten.[2] Dies kann jedoch aus der Sicht des Positionsagenten nicht erzwungen werden, da die Agenten weiterhin ihre Autonomie behalten. Der Mitgliedsagent ist also nicht gezwungen, sich an die Regeln zu halten. Hält er sich nicht an die Regeln, so liegt im technischen Sinne eine Typverletzung vor, die beispielsweise mit der Aufkündigung der Mitgliedschaft durch den Positionsagenten geahndet werden kann.

Im allgemeinen ist der Positionagent für eine geeignete Fehlerbehandlung im Sinne einer Ausnahmebehandlung (execption handling) zuständig. Ein Beispielprokoll zur Kontrolle zeigt die Abbildung 6.7. Im Idealfall werden die Nachrichten einfach weitergeleitet. Nur im Fehlerfall tritt eine Ausnahmebehandlung ein, die hier auch den Mitgliedsagenten einbezieht.

6.4 Ausprägung organisationaler Strukturen

Wie wir gesehen haben, besitzt Teamwork mit den drei Teilaspekten der Teamformation, der Teamplanung und der Ausführungskontrolle eine universelle, von den

[2] Pointiert fomuliert kann man sagen, dass sich ein Organisationsmitglied stets konform mit den Rollenerwartungen, da per Definition nur dann ein Organisationsmitglied ist, wenn er sich an die an ihn gestellten Anforderungen hält.

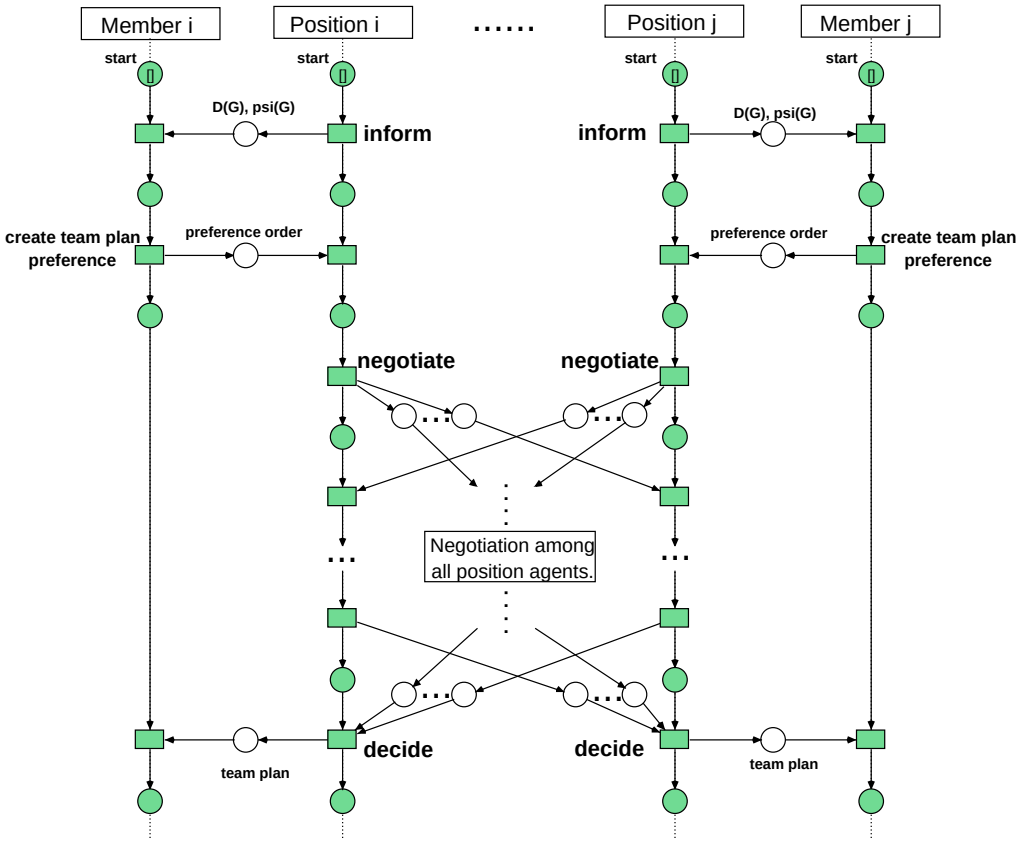


Abbildung 6.6: Protokoll zur Teamplanung

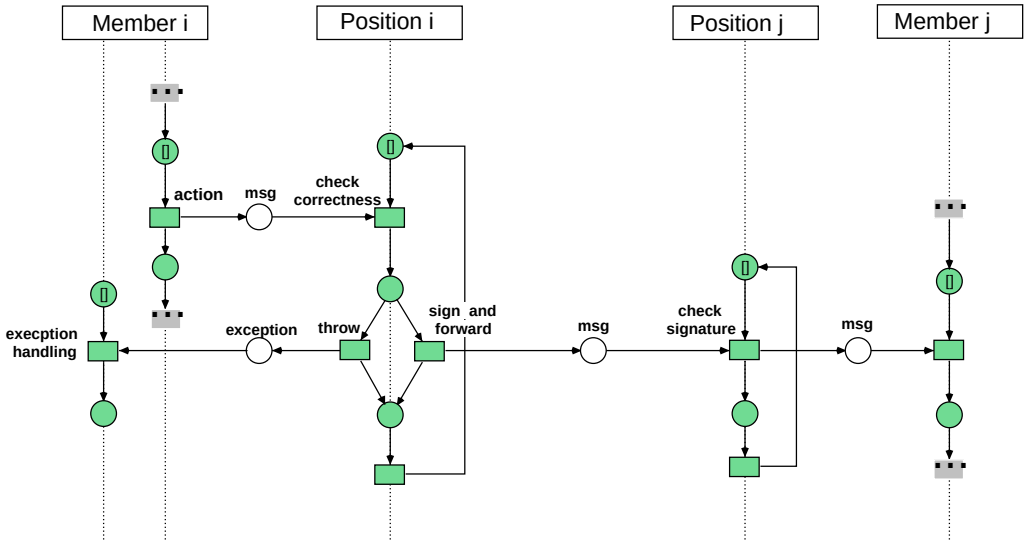


Abbildung 6.7: Protokoll zur Ausführungskontrolle

Planung / Formation	autoritär	teamautoritär	egalitär
Positionen müssen Aufgaben annehmen.	Hierarchien		
Positionen bewerten Aufgaben.		Föderationen	
Positionen können Aufgaben ablehen.			Märkte

Tabelle 6.1: Organisationales Klassifikationsschema

konkreten Aufgaben unabhängige Struktur. Wie wir gesehen haben, lässt dies aber dennoch großen Raum für die Implementation der Position/Mitglied-Interaktion.

Varianten zur Operationalisierung der Teamformation betreffen den Zwang für das Position/Mitglied-Paar, delegierte Aufgaben anzunehmen:

1. Jeder Positionsagent muss jede an ihn delegierte Teilaufgabe bearbeiten.
2. Jeder Positionsagent wird gefragt, ob er bereit ist, eine Teilaufgabe zu bearbeiten. Der delegierende Agent ist aber an eine Ablehnung nicht gebunden.
3. Ein Positionsagent darf sogar die an ihn delegierte Teilaufgabe ablehnen, vorrausgesetzt, dass die Bearbeitung durch einen anderen Agenten garantiert ist.

Varianten zur Operationalisierung der Teamplanung betreffen die Verbindlichkeit der Entscheidung einzelner Agenten für andere bei der Teamplanfindung. Allgemein gilt, dass die Positionsagenten einen Teamplan als Kompromiss aushandeln.

1. Manche Positionsagenten können Vorentscheidungen treffen, die für andere Positionsagenten im folgenden Verhandlungsprozess bindend sind. Diese Möglichkeit zur machtbesetzten, autoritären Planung ergibt sich anhand der Oranisationsstruktur und ist unabhängig vom aktuellen Team.
2. Positionsagenten können für andere Positionsagenten bindende Vorentscheidungen treffen. Diese Autorität ergibt sich aber anhand der Teamstruktur, d.h. dass Positionen nur jenen Vorschriften machen können, die ihnen in der Baumstruktur des Teamnetzes untergeordnet sind.
3. Alle Positionsagenten nehmen gleichberechtigt am Verhandlungsprozess teil. Die Entscheidung wird gemeinschaftlich getroffen.

Die Mitgliedsagenten sind indirekt durch ihren Positionsagenten an der Verhandlung beteiligt. Inwieweit dieser an die Vorlieben des Mitglieds gebunden ist, generiert eine weitere Dimension, die hier aber nicht weiter zur Klassifikation herangezogen werden soll.

Beide Aspekte – Teamformation und -planung – stellen unabhängige Dimensionen der Teamprozesse und damit auch der Organisiertheit dar.

Es ergibt sich somit das zweidimensionale Schema aus Tabelle 6.1 zur Klassifikation von Teamprozessen. Jeder Teamprozess besitzt im Prinzip die gleichen Grundstruktur, bestehend aus Formation, Planung und Durchführung. Die jeweilige Ausprägung in Gestalt der oben beschriebenen Operationalisierungen unerscheiden die Prozesse des Teamworks jedoch charakteristisch.

In (Köhler und Wester-Ebbinghaus, 2007a,b) wurden drei typische Formen dieser Formation/Teamplanung-Interaktion identifiziert: Betriebe als Idealform hierarchischer, autoritärer Teamprozesse, virtuelle Organisationen (auch: Föderationen) als Idealform kongregierter und kooperierender Teamprozesse und Märkte als Idealform egalitärer, verhandelnder Teamprozesse. Diese Formen bilden die Hauptdiagonalelemente in der Klassifikationsmatrix aus Tabelle 6.1.

Zusammenfassung

In diesem Abschnitt haben wir gezeigt, dass sich Agentensysteme geeignet als Interaktion von formaler Organisation und ihren Mitgliedern darstellen lassen. Dies erweitert die bisherige Darstellung der formalen Organisation um den Aspekt, dass es die Organisationsmitglieder sind, welche die organisationalen Prozesse ausführen und mit Leben füllen. Die Teamprozesse werden also nicht ausschließlich von den Positionen erbracht, sondern vielmehr von dem aus Position und Organisationsmitglied bestehenden Paar. Dieses Paar besteht dabei gleichermaßen aus zwei analytisch trennbaren Komponenten, obgleich praktisch keine der beiden Komponenten jemals allein auftreten würde. Daher sind in die Teamprozesse der Formation, der Planung und der Planausführung stets Positionen und Mitglieder involviert, mit der Besonderheit, dass Mitglieder ausschließlich durch ihre jeweiligen Positionen wirksam werden können. Letzteres drückt sich insbesondere in den Interaktionsprotokollen der Teamprozesse aus. Außerdem zeigt sich, dass die konkreten Ausprägungen der Teamformation und -planung ein hinreichend reiches Spektrum organisationaler Strukturen aufspannt, um darin die aus der Organisationstheorie bekannten Strukturen der Hierarchie, der Föderation oder des Marktes wiederzuerkennen. Es zeigt sich somit, dass das hier gezeigte Modell reich genug ist, um die bereits bekannten und praktisch erpobten Organisationsformen direkt zu modellieren.

Literaturverzeichnis

[Aalst 1997] Aalst, Wil van d.: Verification of Workflow Nets. In: (Azeme und Balbo, 1997), S. 407–426

[Aalst 1998] Aalst, Wil van d.: The application of Petri nets to workflow management. In: *Journal of Circuits, Systems and Computers* 8 (1998), Nr. 1, S. 21–66

[Alonso u. a. 2003] Alonso, Gustavo ; Casati, Fabio ; Kuno, Harumi ; Machiraju, Vijay: *Web Services*. Springer-Verlag, 2003

[Azeme und Balbo 1997] Azeme, Pierre (Hrsg.) ; Balbo, Gianfranco (Hrsg.): *Proceeding of the 18th International Conference on Application and Theory of Petri Nets*. Bd. *1248*. Springer-Verlag, 1997. (Lecture Notes in Computer Science)

[Baader u. a. 2003] Baader, F. (Hrsg.) ; Calvanese, D. (Hrsg.) ; McGuinness, D. (Hrsg.) ; Nardi, D. (Hrsg.) ; Patel-Schneider, P.F. (Hrsg.): *Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003

[Baeten 1990] Baeten, J. C. M. (Hrsg.): *Cambridge Tracts in Theoretical Computer Science*. Bd. 17: *Applications of Proces Algebra*. Cambridge University Press, 1990

[Barringer u. a. 1990] Barringer, H. ; Fisher, M. ; Gabbay, D. ; Gough, G. ; Owens, R.: METATEM: a framework for programming in temporal logic. In: *REX workshop: Proceedings on Stepwise refinement of distributed systems: models, formalisms, correctness*. New York, NY, USA : Springer-Verlag New York, Inc., 1990, S. 94–129. – ISBN 0-387-52559-9

[Bellifemine u. a. 2001] Bellifemine, Fabio ; Poggi, Agostino ; Rimassa, Giovanni: Developing multi-agent systems with JADE. In: Castelfranchi, Cristiano (Hrsg.) ; Lespérance, Yves (Hrsg.): *Seventh International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000)* Bd. 1986, Springer-Verlag, 2001, S. 89–103

[Billington u. a. 1998] Billington, J. ; Du, B.B. ; Farrington, M.: Modelling and Analysis of Multi-Agent Communication Protocols using CP-nets. In: *Engineering Mathematics and Applications (EMAC'98)*, Juli 1998, S. 119–122

[Booch 1992] Booch, Grady: *Object-oriented analysis and design: with applications*. Benjamin/Cummings, 1992

[Bordini u. a. 2007] Bordini, Rafael H. ; Hübner, Jomi F. ; Wooldridge, Michael: *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley, 2007

[Bouhoula u. a. 2000] Bouhoula, Adel ; Jouannaud, Jean-Pierre ; Meseguer, José: Specification and Proof in Membership Equational Logic. In: *Theoretical Computer Science* 236 (2000), S. 35–132

[Bradfield 1992] Bradfield, Julian C.: *Verifying temporal properties of systems*. Birkhäuser, 1992

[Bratman 1987] Bratman, M. E.: *Intentions, Plans, and Practical Reason*. Cambridge : Harvard University Press, 1987

[Bresciani u. a. 2004] Bresciani, P. ; Giorgini, P. ; Giunchiglia, F. ; Mylopoulos, J. ; Perini, A.: Tropos: An Agent-Oriented Software Development Methodology. In: *Journal of Autonomous Agents and Multi-Agent Systems* 8 (2004), S. 203–236

[Brooks 1990] Brooks, Rodney A.: A robust layered control system for a mobile robot. (1990), S. 2–27

[Cabac u. a. 2003] Cabac, Lawrence ; Moldt, Daniel ; Rölke, Heiko: A Proposal for Structuring Petri Net-Based Agent Interaction Protocols. In: Aalst, W. v. d. (Hrsg.) ; Best, E. (Hrsg.): *International Conference on Application and Theory of Petri Nets 2003* Bd. 2679, Springer-Verlag, 2003, S. 102–120

[Carley und Gasser 1999] Carley, Kathleen M. ; Gasser, Les: Computational organisation theory. In: (Weiß, 1999), S. 229–330

[Carriero und Gelernter 1989] Carriero, N. ; Gelernter, D.: Linda in context. In: *Communications of the ACM* 32 (1989), S. 444–458

[Castelfranchi u. a. 1999] Castelfranchi, C. ; Dignum, F. ; Jonker, C. ; Treur, J.: Deliberate Normative Agents: Principles and Architecture. In: *Proceedings of the Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL 99)*, 1999

[Castelfranchi 1995] Castelfranchi, Christiano: Commitments: from individual intentions to groups and organisations. In: *First International Conference on Multi Agent Systems*, AAAI Press and MIT Press, 1995, S. 41–48

[Castelfranchi 2000] Castelfranchi, Christiano: Engineering social order. In: Omicini, A. (Hrsg.) ; Tolksdorf, R. (Hrsg.) ; Zambonelli, F. (Hrsg.): *Engineering Societies in the Agents World. First International Workshop, ESAW 2000, Berlin, Germany* Bd. 1972, Springer-Verlag, 2000, S. 1–18

[Castelfranchi und Conte 1995a] Castelfranchi, Christiano ; Conte, Rosaria: *Cognitive and Social Action*. UCL Press, 1995

[Castelfranchi und Conte 1995b] Castelfranchi, Christiano ; Conte, Rosaria: Understanding the Functions of Norms in Social Groups Through Simulation. In: Gilbert, N. (Hrsg.) ; Conte, R. (Hrsg.): *Artificial Societies: The Computer Simulation of Social Life*. UCL Press: London, 1995, S. 252–267

[Castelfranchi und Conte 1996] Castelfranchi, Christiano ; Conte, Rosaria: Distributed artificial intelligence and social science: Critical issues. In: O'Hare, G. M. P. (Hrsg.) ; Jennings, N. R. (Hrsg.): *Foundations of Distributed Artificial Intelligence*, Wiley, 1996, S. 527–542

[Castelfranchi und Conte 1995c] Castelfranchi, Cristiano ; Conte, Rosaria: Norms as Mental Objects - From Normative Beliefs to Normative Goals. In: *5th European Workshop on Modelling Autonomous Agents (MAAMAW '93)* Bd. 957, Springer, 1995, S. 186–196

[Castelfranchi und Conte 1999] Castelfranchi, Cristiano ; Conte, Rosaria: From Conventions to Prescriptions – Towards an Integrated View of Norms. In: *Artificial Intelligence and Law* 7 (1999), Nr. 4, S. 323–340

[Chainbi u. a. 1996] Chainbi, W. ; Hanachi, C. ; Sibertin-Blanc, C.: The Multi-agent Prey/Predator problem: A Petri net solution. In: Borne, P. (Hrsg.) ; Gentina, J.C. (Hrsg.) ; Craye, E. (Hrsg.) ; Khattabi, S., El (Hrsg.): *Proceedings of the Conference on Computational Engineering in Systems Applications (CESA'96)*, IEEE Society Press, 1996, S. 291–299

[Chandy und Misra 1989] Chandy, Mani ; Misra, Jayadev: *Parallel programm design: a foundation*. Adison-Wesley, 1989

[Clarke u. a. 1989] Clarke, Edmund M. C. ; Long, David E. ; McMillan, Keneth L.: Compositional Model Checking. In: *Proc LICS'89*, IEEE Computer Society Press, 1989, S. 353–363

[Cohen und Levesque 1991] Cohen, Phil R. ; Levesque, Hector J.: Teamwork. In: *Nous, Special Issue on Cognitive Science and Artifical Intelligence* 25 (1991), Nr. 4, S. 487–512

[Cost u. a. 1999] Cost, R. S. ; Chen, Ye ; Finin, T. ; Labrou, Y. ; Peng, Y.: Modeling agent conversation with colored Petri nets. In: *Workshop on specifying and implementing conversation policies (Autonomous agents '99)*, Springer-Verlag, 1999, S. 565–579

[DeLoach u. a. 2001] DeLoach, Scott A. ; Wood, Mark F. ; Sparkman, Clint H.: Multiagent Systems Engineering. In: *International Journal of Software Engineering and Knowledge Engineering* 11 (2001), Nr. 3, S. 231–258

[Desel u. a. 1992] Desel, Jörg ; Gomm, Dominik ; Kindler, Ekkart ; Walter, Rolf ; Paech, Barbara: Bausteine eines kompositionalen Beweiskalküls für netzmodellierte Systeme / Technische Universität München. 1992. – Forschungsbericht. Sonderforschungsbereich 342: Methoden und Werkzeuge für die Nutzung paralleler Rechnerarchitekturen

[Dignum u. a. 2004] Dignum, Virginia ; Vázquez-Salceda, Javier ; Dignum, Frank: OMNI: Introducing Social Structure, Norms and Ontologies into Agent Organizations. In: *Programming Multi-Agent Systems (PROMAS)*, 2004, S. 181–198

[Durfee und Lesser 1991] Durfee, Edmund H. ; Lesser, Victor R.: Partial Global Planning: A Coordination Framework for Distributed Hypothesis Formation. In: *IEEE Transactions on Systems, Man, and Cybernetics* 21 (1991), September, Nr. 5, S. 1167–1183

[Ehrig und Mahr 1985] Ehrig, Hartmut ; Mahr, Bernd: *Fundamentals of algebraic Specification*. Springer-Verlag, 1985 (EATCS Monographs on TCS)

[Emerson 1990] Emerson, Allen: *Temporal and modal logic*. Bd. B. MIT press, 1990

[Emerson und Clarke 1982] Emerson, Allen ; Clarke, Edmund M.: Using branching time temporal logics to synthesize synchronisation sceletons. In: *Scientific Compututer Programming* 2 (1982), S. 241–266

[Esparza 1994] Esparza, Javier: Model checking using net unfoldings. In: *Science of Computer Programming* 23 (1994), Nr. 2, S. 151–195

[Ferber u. a. 2003] Ferber, Jacques ; Gutknecht, Olivier ; Michel, Fabien: From Agents to Organizations: An Organizational View of Multi-agent Systems. In: Giorgini, Paolo (Hrsg.) ; Müller, Jörg P. (Hrsg.) ; Odell, James (Hrsg.): *Agent-Oriented Software Engineering IV* Bd. 2935, 2003, S. 214–230

[Ferber 1999] Ferber, Jaques: *Multi-Agent System: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, 1999

[Ferguson 1992] Ferguson, I. A.: Towards an Architecture for adaptive, rational, mobile agents. In: *Decentralized Artificial Intelligence* Bd. 3, Elsevier Science, 1992, S. 249–262

[Fiorino und Tessier 1998] Fiorino, H. ; Tessier, C.: Agent cooperation: a Petri net based model. In: Demazeau, Yves (Hrsg.): *Int. Conference on Multi-Agent Systems (ICMAS'98)*, 1998, S. 425–426

[FIPA 1998] FIPA: FIPA 97 Specification, Part 2 - Agent Communication Language. `http://www.fipa.org` : Foundation for Intelligent Physical Agents, Oktober 1998. – Forschungsbericht

[Floyd 1993] Floyd, Christiane: STEPS - A Methodical Approach to Participatory Design. In: *Communications of the ACM* 36 (1993), Nr. 4, S. 83

[Friedberg 1995] Friedberg, Erhard: *Ordnung und Macht. Dynamiken organisierten Handelns*. Frankfurt and New York : Campus Verlag, 1995

[Gabbay u. a. 1980] Gabbay, Dov ; Pnueli, Amir ; Shelah, Saharon ; Stavi, Jonathan: On The Temporal Analysis of Fairness. In: *Proceedings of the 7th Annual ACM Symposium on Principles of Programming Languages*, 1980, S. 163–173

[Gasser 1991] Gasser, Les: Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems Semantics. In: *Artificial Intelligence* 47 (1991), S. 107–138

[Genrich und Lautenbach 1981] Genrich, Hartmann J. ; Lautenbach, Kurt: System modelling with high-level Petri Nets. In: *Theoretical Foundations of Computer Science* 13(1) (1981), S. 109–136

[Giacomo u. a. 2000] Giacomo, Giuseppe D. ; Lespérance, Yves ; Levesque, Hector J.: ConGolog, a concurrent programming language based on the situation calculus. In: *Artificial Intelligence* 121 (2000), Nr. 1-2, S. 109–169

[Glaser und Morignot 1997] Glaser, Norbert ; Morignot, Philippe: The Reorganization of Societies of Autonomous Agents. In: Boman, Magnus (Hrsg.) ; Velde, Walter V. de (Hrsg.): *Multi-Agent Rationality, 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World* Bd. 1237, Springer-Verlag, 1997, S. 98–111

[Godefroid 1996] Godefroid, Patrice: *Partial-order methods for the verification of concurrent systems: an approach to the state-explosion problem.* Springer-Verlag, 1996

[Gois u. a. 1998] Gois, Gustavo M. ; Perkusich, Angelo ; Figueiredo, Jorge C. A. de ; Costa, Evandro B.: Towards a Multi-agent Interactive Learning Environment Oriented to the Petri Net Domain. In: *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC'98), 11-14 October 1998, San Diego, USA*, Oktober 1998, S. 250–255

[Gottschalk 2000] Gottschalk, Karl: Web Services architecture overview / IBM developerWorks. 2000. – Whitepaper

[Griffel 1998] Griffel, Frank: *Componentware: Konzepte und Techniken eines Softwareparadigmas.* dpunkt Verlag, 1998

[Knowledge Sharing Initiative External Interfaces Working Group 1993] Group, The D. Knowledge Sharing Initiative External Interfaces Working: Specification of the KQML Agent-Communication Language / DARPA. Juni 1993. – DRAFT

[Grumberg und Long 1994] Grumberg, Orna ; Long, David: Model checking and modular verification. In: *ACM Transactions on Programming Languages and Systems* 16 (1994), Nr. 3, S. 843–871

[Hameurlain u. a. 1999] Hameurlain, N. ; Hanachi, C. ; Sibertin-Blanc, C.: Mobile agents behaviours: from declaratiove specifications to implementations. In: Klusch, Matthias (Hrsg.) ; Shehory, Onn M. (Hrsg.) ; Weiss, Gerhard (Hrsg.): *Proceedings of the third Workshop on Cooperative Information Agents (CIA'99)*, Springer-Verlag, 1999 (Lecture notes in artificial intelligence), S. 196–207

[Hannoun u. a. 2000] Hannoun, Mahdi ; Boissier, Olivier ; Sichman, Jaime S. ; Sayettat, Claudette: MOISE: An Organizational Model for Multi-agent Systems. In: *IBERAMIA-SBIA '00: Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI*, Springer-Verlag, 2000, S. 156–165

[Harel 1984] Harel, David: Dynamic Logic. In: Gabbay, D. (Hrsg.) ; Guenther, F. (Hrsg.): *Handbook of Philosophical Logic, Volume II — Extensions of Classical Logic.* D. Reidel Publishing Company: Dordrecht, The Netherlands, 1984, S. 497–604

[van Hee u. a. 2003] Hee, Kees van ; Sidorova, Natalia ; Voorhoeve, Marc: Soundness and Separability of Workflow Nets in the Stepwise Refinement Approach. In: *Proceedings of the 24th International Conference on Applications and Theory of Petri Nets (ICATPN 2003)* Bd. 2679, Springer-Verlag, 2003, S. 337–356

[Hennessy und Milner 1980] Hennessy, Matthew ; Milner, Robin: On observing nondeterminism and concurrency. In: Bakker, J.W. de (Hrsg.) ; Leeuwen, J. van (Hrsg.): *Proc. of the Seventh International Colloquium on Automata Languages and Programming (ICALP)* Bd. 85, Springer-Verlag, 1980, S. 299–309

[Hoffmann u. a. 2005] Hoffmann, Kathrin ; Ehrig, Hartmut ; Mossakowski, Till: High-Level Nets with Nets and Rules as Tokens. In: *Application and Theory of Petri Nets and Other Models of Concurrency* Bd. 3536, Springer-Verlag, 2005, S. 268 – 288

[Holvoet 1995] Holvoet, Tom: Agents and Petri Nets. In: *Petri Net Newsletter* 49 (1995), Oktober, S. 3–8

[Horling und Lesser 2005a] Horling, Bryan ; Lesser, Victor: A Survey of Multi-Agent Organizational Paradigms. In: *The Knowledge Engineering Review* 19 (2005), Nr. 4, S. 281–316

[Horling und Lesser 2005b] Horling, Bryan ; Lesser, Victor: Using ODML to Model Organizations for Multi-Agent Systems. In: *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2005).* Compiegne, France : IEEE Computer Society, September 2005, S. 72–80

[Hughes und Cresswell 1984] Hughes, George E. ; Cresswell, Maxwell J.: *A companion to modal logic*. Methuen, 1984

[Jennings 1993] Jennings, Nicholas R.: Commitments and Conventions: The Foundation of Coordination in Multi-Agent Systems. In: *The Knowledge Engineering Review* 8 (1993), Nr. 3, S. 223–250

[Jennings 1996] Jennings, Nicholas R.: Coordination Techniques for Distributed Artificial Intelligence. In: O'Hare, G. M. P. (Hrsg.) ; Jennings, N. R. (Hrsg.): *Foundations of Distributed Artificial Intelligence*, Wiley, 1996, S. 187–210

[Jennings 2000] Jennings, Nicholas R.: On agent-based software engineering. In: *Artificial Intelligence* 117 (2000), S. 277–296

[Jensen 1992] Jensen, Kurt: *Coloured Petri nets, Basic Methods, Analysis Methods and Practical Use*. Springer-Verlag, 1992 (EATCS monographs on theoretical computer science)

[Jensen und Rozenberg 1991] Jensen, Kurt (Hrsg.) ; Rozenberg, Grzegorz (Hrsg.): *High-level Petri nets: theory and application*. Berlin : Springer-Verlag, 1991

[Kennedy und Eberhart 2001] Kennedy, James ; Eberhart, Russell C.: *Swarm intelligence*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2001. – ISBN 1-55860-595-9

[Kephart und Chess 2003] Kephart, Jeffrey O. ; Chess, David M.: The Vision of Autonomic Computing. In: *IEEE Computer* 36 (2003), Nr. 1, S. 41–50

[Kindler 1995] Kindler, Ekkart: *Modularer Entwurf verteilter Systeme mit Petrinetzen*. Bertz Verlag, 1995

[Kindler 1997] Kindler, Ekkart: A Compositional Partial Order Semantics for Petri Net Components. In: (Azeme und Balbo, 1997)

[Kirn und Gasser 1998] Kirn, Stefan ; Gasser, Les: *Organizational Approaches to Coordination in Multi-Agent Systems*. 1998

[Koestler 1967] Koestler, Arthur: *The Ghost in the Machine*. London : Arkana, 1967

[Köhler 2007] Köhler, Michael: A Formal Model of Multi-Agent Organisations. In: *Fundamenta Informaticae* 79 (2007), Nr. 3-4, S. 415 – 430

[Köhler u. a. 2006] Köhler, Michael ; Moldt, Daniel ; Ortmann, Jan: Dynamic Service Composition: A Petri-Net Based Approach. In: Manolopoulos, Y. (Hrsg.) ; Filipe, J. (Hrsg.) ; Constantopoulos, P. (Hrsg.) ; Cordeiro, J. (Hrsg.): *Conference on Enterprise Information Systems: Databases and Information Systems Integration (ICEIS 2006)*, 2006, S. 159–165

[Köhler u. a. 2001] Köhler, Michael ; Moldt, Daniel ; Rölke, Heiko: Modeling the Behaviour of Petri Net Agents. In: Colom, J. M. (Hrsg.) ; Koutny, M. (Hrsg.): *International Conference on Application and Theory of Petri Nets* Bd. 2075, Springer-Verlag, 2001, S. 224–241

[Köhler und Ortmann 2005] Köhler, Michael ; Ortmann, Jan: Formal Aspects for service modelling based on high-level Petri Nets. In: Mohammadian, M. (Hrsg.): *International Conference on Intelligent Agents, Web Technologies and Internet Commerce (IAWTIC 2005)*, 2005

[Köhler und Wester-Ebbinghaus 2007a] Köhler, Michael ; Wester-Ebbinghaus, Matthias: Organizational Models and Multi-Agent System Deployment. In: *Multi-Agent Systems and Applications V* Bd. 4696, Springer-Verlag, 2007, S. 307–309

[Köhler und Wester-Ebbinghaus 2007b] Köhler, Michael ; Wester-Ebbinghaus, Matthias: Petri Net-Based Specification and Deployment of Organizational Models. In: *International Workshop on Petri Nets and Software Engineering 2007*, 2007, S. 67–81

[Lamport 1977] Lamport, Leslie: Proving the corretness of multiprocess programs. In: *IEEE Transactions on Software Engineering* 3 (1977), Nr. 2, S. 125–143

[Lamport 1994] Lamport, Leslie: The temporal logics of actions. In: *ACM Transactions on Programming Languages and Systems* 16 (1994), Nr. 3, S. 872–923

[Langer 2005a] Langer, Roman: *Anerkennung und Vermögen: Eine sozialtheoretische Analyse der Selbstorganisation in und von Bildungsinstitutionen*. Münster : Monsenstein & Vannerdat, 2005

[Langer 2005b] Langer, Roman: *Hinter den Spiegeln universitärer Governance*. Münster : Lit-Verlag, 2005 (Wirtschaft – Arbeit – Technik). – Unter Mitarbeit von Daniela Spresny.

[Larsem und Thomson 1991] Larsem, Kim G. ; Thomson, Bent: Partial specifications and compositional specification. In: *Theoretical Computer Science* 88 (1991), S. 15–32

[Levesque und Lakemeyer 2000] Levesque, Hector ; Lakemeyer, Gerhard: *The logic of knowledge bases*. Cambridge, Massachusetts : MIT Press, 2000

[Loeckx u. a. 1996] Loeckx, Jacques ; Ehrich, Hans-Dieter ; Wolf, Markus: *Specification of Abstract Data Types*. New York, Leipzig : Wiley & Sons and B.G. Teubner, 1996

[Manna und Pnueli 1995] Manna, Zohar ; Pnueli, Amir: *Temporal verification of reactive systems*. Springer-Verlag, 1995

[Maude 1999] Clavel, Manuel ; Durán, Francisco ; Eker, Steven ; Lincoln, Patrick ; Martí-Oliet, Narciso ; Meseguer, José ; Quesada, José: *Maude: Specification and Programming in Rewriting Logic. Maude System documentation*. `http://maude.cs.uiuc.edu/maude1/`: , 1999

[McMillan 1993] McMillan, Keneth L.: *Symbolic Model Checking*. Kluwer Academic, 1993

[Meseguer 1997] Meseguer, José: Membership algebra as a logical framework for equational specification. In: Parisi-Presicce, Francesco (Hrsg.): *Workshop on Algebraic Development Techniques (WADT'97)* Bd. 1376, Springer-Verlag, 1997, S. 18–61

[Milner 1989] Milner, Robin: *Communication and Concurrency*. Prentice Hall, 1989

[Moldt und Wienberg 1997] Moldt, Daniel ; Wienberg, Frank: Multi-Agent-Systems Based on Coloured Petri Nets. In: (Azeme und Balbo, 1997), S. 82–101

[Müller und Pischel 1994] Müller, Jörg P. ; Pischel, Markus: An architecture for dynamically interacting agents. In: *Journal of Intelligent and Cooperative Information Systems* 3 (1994), Nr. 1, S. 25–45

[Müller 1993] Müller, Jürgen: *Verteilte künstliche Intelligenz: Methoden und Anwendungen*. B-I-Wiss.-Verlag, 1993

[Nagendra u. a. 1996] Nagendra, Prasad ; Garvey, Alan ; Decker, Keith ; Lesser, Victor: Exploring Organizational Designs with TAEMS: A case study of distributed data processing. In: *Second International Conference on Multi-Agent Systems* (1996), January, S. 283–290

[Nierstrasz 1993] Nierstrasz, Oscar: Composing active objects. In: *Research directions in concurrent object-oriented programming*. MIT Press, 1993, Kap. 5

[OASIS 1993–2007] : *Organization for the Advancement of Structured Information Standards*. www.oasis-open.org. 1993-2007

[Ogbuji und Ouellet 2002] Ogbuji, Uche ; Ouellet, Roxane: *DAML Reference*. online: `http://www.xml.com/pub/a/2002/05/01/damlref.html`. 2002

[Ossowski 1999] Ossowski, Sascha: *Lecture Notes in Computer Science*. Bd. 1535: *Co-ordination in Artificial Agent Societies: social structures and its implications for autonomous problem-solving agents*. Springer-Verlag, 1999

[Padberg u. a. 1998] Padberg, J. ; Gajewsky, M ; Ermel, C.: Rule-based refinement of high-level nets preserving safety properties. In: Astesiani, E. (Hrsg.): *Proceedings of ETAPS-FASE: Fundamental approaches to software engineering*, Springer-Verlag, 1998 (LNCS 1382), S. 221–238

[Padberg u. a. 1995] Padberg, J. ; Mahr, E. ; Ribeiro, L.: Algebraic high-level net transformation systems. In: *Mathematical structures in computer science* (1995), Nr. 5, S. 217–256

[Panzarasa und Jennings 2001] Panzarasa, Pietro ; Jennings, Nicholas: The organisation of sociality: A manifesto for a new science of multiagent systems. In: *Proceedings of the Tenth European Workshop on Multi-Agent Systems (MAAMAW01)*, 2001

[Panzarasa u. a. 2002] Panzarasa, Pietro ; Jennings, Nicholas R. ; Norman, Timothy J.: Formalizing Collaborative Decision-making and Practical Reasoning in Multi-agent Systems. In: *Journal of Logic and Computation* 12 (2002), Nr. 1, S. 55–117

[Park 1980] Park, D. M. R.: *Lecture Notes in Computer Science*. Bd. 104: *Concurrency and Automata on Infinte Sequences*. Springer-Verlag, 1980

[Park 1976] Park, David: Finiteness is Mu-Ineffable. In: *Theoretical Computer Science* 3 (1976), Nr. 2, S. 173–181

[Pnueli und Lichtenstein 1985] Pnueli, Amir ; Lichtenstein, Orna: Checking that Finite State Concurrent Programs Satisfy their Linear Specification. In: *Conference record of the 14th ACM Symposium on Principles of Programming Languages (POPL)*, 1985, S. 97–107

[Prietula u. a. 1998] Prietula, Michael J. (Hrsg.) ; Carley, Kathleen M. (Hrsg.) ; Gasser, Les (Hrsg.): *Simulating Organisations. Computational Models of Institutions and Groups*. AAAI/MIT-Press, 1998

[Queille und Sifakis 1981] Queille, Jean-Pierre ; Sifakis, Joseph: *Specification and verification of concurrent systems in Cesar*. 1981

[Rao 1996] Rao, Anand S.: AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In: Hoe, Rudy van (Hrsg.): *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, 1996

[Rao und Georgeff 1991] Rao, Anand S. ; Georgeff, Michael P.: Modeling Rational Agents within a BDI-Architecture. In: Allen, James (Hrsg.) ; Fikes, Richard (Hrsg.) ; Sandewall, Erik (Hrsg.): *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91)*, Morgan Kaufmann, 1991, S. 473–484

[dos Reis Coutinho u. a. 2005] Reis Coutinho, Luciano dos ; Sichman, Jaime S. ; Boissier, Olivier: Modeling organization in MAS: a comparison of models. In: Choren, R. (Hrsg.) ; Silva, V. (Hrsg.): *Proc. of the 1st. Workshop on Software Engineering for Agent-Oriented Systems (SEAS'05)*, 2005

[Reisig 1988] Reisig, Wolfgang: Temporal logics and causality in concurrent systems – Petri net models of distributed algoritms. In: Vogt, F. H. (Hrsg.): *International Conference on Concurrency 88* Bd. 335. Springer-Verlag, 1988

[Reisig 1991] Reisig, Wolfgang: Petri nets and algebraic specifications. In: *Theoretical Computer Science* 80 (1991), S. 1–34

[Saam 2001] Saam, Nicole J.: Social norms for co-operative agents. In: Saam, N. J. (Hrsg.) ; Schmidt, B. (Hrsg.): *Cooperative Agents: Applications in the social sciences* Bd. 32. Dordrecht, Boston, London : Kluwer Academic Publishers, 2001, S. 39–56

[Salomaa 1987] Salomaa, Arto: *Formal languages*. San Diego, CA, USA : Academic Press Professional, 1987

[Schillo 2003] Schillo, Michael: Self-Organization and Adjustable Autonomy: Two Sides of the Same Coin? In: *Connection Science* 14 (2003), Nr. 4, S. 345–360

[Schillo u. a. 2000] Schillo, Michael ; Fischer, Klaus ; Klein, Christof: The Micro-Macro Link in DAI and Sociology. In: Moss, S. (Hrsg.) ; Davidsson, P. (Hrsg.): *Second International Workshop on Multi-Agent Based Simulation* Bd. 1979, Springer-Verlag, 2000, S. 133–148

[Schillo und Spresny 2005] Schillo, Michael ; Spresny, Daniela: Organization: The Central Concept for Qualitative and Quantitative Scalability. In: Fischer, Klaus (Hrsg.) ; Florian, Michael (Hrsg.) ; Malsch, Thomas (Hrsg.): *Socionics: Sociability of Complex Social Systems* Bd. 3413, Springer-Verlag, 2005

[Schmidt-Schauß und Smolka 1991] Schmidt-Schauss, M. ; Smolka, G.: Attributive concept descriptions with complements. In: *Articial Intelligence* 48 (1991), Nr. 1, S. 1–26

[Schumacher 2001] Schumacher, Michael: *Objective coordination in multi-agent system engineering: design and implementation*. Springer-Verlag, 2001

[Searle 1970] Searle, John R.: *Speech acts: An essay in the philosophy of Language*. Cambridge University Press, 1970

[Shankar 1998] Shankar, Udaya: Machine-assisted verifcation using theorem proving and model checking. In: Broy, M. (Hrsg.): *Mathematical methods in programm development*, Springer-Verlag, 1998

[Shoham 1990] Shoham, Yoav: Agent-oriented programming / Stanford, Calif.: Department of Computer Science, Stanford University. 1990. – Forschungsbericht

[Shoham und Tennenholtz 1994] Shoham, Yoav ; Tennenholtz, Moshe: On social laws for artificial agent societies: off-line design. In: *Artificial Intelligence* 72 (1994), Nr. 1-2, S. 231–252

[Smith u. a. 2004] Smith, Michael K. ; Welty, Chris ; McGuinness, Deborah L.: *OWL Web Ontology Language Guide. W3C Recommendation*. `http://www.w3.org/TR/owl-guide/`. 2004

[Smith 1977] Smith, Reid G.: The contract net: A formalism for the control of distributed problem solving. In: *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*, 1977

[Sozionik 1998] SPP Sozionik: *Sozionik: Erforschung und Modellierung künstlicher Sozialität*. http://www.tu-harburg.de/tbg/SPP/spp-antrag.html. 1998. – Antragstext zum DFG-Schwerpunktprogramm *Sozionik*.

[Studer u. a. 2003] Studer, Rudi ; Hotho, Andreas ; Stumme, Gerd ; Volz, Raphael: Semantic Web – State of the Art and Future Directions. In: *Künstliche Intelligenz* 3 (2003), S. 5–9

[Thomas 1990] Thomas, Wolfgang: Automata on infinite objects. In: Leeuwen, J. van (Hrsg.): *Handbook of Theoretical Computer Science: Formal Models and Semantics*. Elsevier, 1990, S. 133–192

[Turner und Jennings 2001] Turner, Phillip J. ; Jennings, Nicholas R.: Improving the Scalability of Multi-agent Systems. In: *Proceedings of the First International Workshop on Infrastructure for Scalable Multi-Agent Systems* Bd. 1887, Springer-Verlag, 2001, S. 246ff.

[Ullman-Margalit 1977] Ullman-Margalit, Edna: *The Emergence of Norms*. Clarendon Press, 1977

[v. Lüde, Moldt und Valk 2003] Lüde, R. v. ; Moldt, D. ; Valk, R. ; Köhler, M. ; Langer, R. ; Rölke, H. ; Spresny, D.: *Sozionik: Modellierung soziologischer Theorie*. Münster : Lit-Verlag, 2003 (Wirtschaft – Arbeit – Technik). – URL `http://www.lit-verlag.de/isbn/3-8258-5980-0`

[Valmari 1990] Valmari, Antti: Compositional State Space Generation. In: Rozenberg, G. (Hrsg.): *Proceedings of the 11th International Conference on Application and Theory of Petri Nets* Bd. 524, Springer-Verlag, 1990, S. 43–62

[Valmari 1994] Valmari, Antti: *Lecture Notes in Computer Science*. Bd. 815: *Compositional Analysis of place-bordered Subnets*. Springer-Verlag, 1994

[Vardi und Wolper 1994] Vardi, Moshe Y. ; Wolper, Pierre: Reasoning About Infinite Computations. In: *Information and Computation* 115 (1994), 15 , Nr. 1, S. 1–37

[Walker und Wooldridge 1995] Walker, A. ; Wooldridge, M. J.: Understanding the Emergence of Conventions in Multi-Agent Systems. In: *Proceedings of the 1st International Conference on Multiagent Systems (ICMAS'95)*, 1995

[Weiß 1999] Weiss, Gerhard (Hrsg.): *Multiagent systems: A modern approach to Distributed Artificial Intelligence*. MIT Press, 1999

[Weiß und Jakob 2004] Weiss, Gerhard ; Jakob, Ralf: *Agentenorientierte Softwareentwicklung: Methoden und Tools*. Springer-Verlag, 2004

[Wooldridge 2000] Wooldridge, Michael: *Reasoning about Rational Agents*. Cambridge, Massachussetts/London : MIT Press, 2000 (Intelligent robotics and autonomous agents)

[Wooldridge und Jennings 1995] Wooldridge, Michael J. ; Jennings, Nicholas R.: Agent Theories, Architectures, and Languages: a Survey. In: Wooldridge, M. J. (Hrsg.) ; Jennings, N. R. (Hrsg.): *Intelligent Agents*, Springer-Verlag, 1995, S. 1–22

[Zambonelli u. a. 2003] Zambonelli, Franco ; Jennings, Nicholas R. ; Wooldridge, Michael: Developing multiagent systems: The Gaia methodology. In: *ACM Trans. Softw. Eng. Methodol.* 12 (2003), Nr. 3, S. 317–370