Process Landscaping
Eine Methode zur Modellierung und Analyse verteilter Softwareprozesse

Dissertation

zur Erlangung des Grades eines

DOKTORS DER NATURWISSENSCHAFTEN

der Universität Dortmund
am Fachbereich Informatik

von

Ursula Wellen

Dortmund
2003

Zusammenfassung

Die Betrachtung komplexer Softwareentwicklungsprozesse erfordert nicht nur die Berücksichtigung der traditionellen Entwicklungsphasen wie Anforderungsdefinition, Analyse, Entwurf, Implementation und Test, sie verlangt eine ganzheitliche Betrachtung aller Aktivitäten rund um ein Softwareentwicklungsprojekt. Mit einer solchen ganzheitlichen Betrachtung wird eine Reihe unterschiedlicher Ziele verfolgt. Dazu zählen vor allem die Dokumentation und Analyse von Softwareentwicklungsprozessen zum einen für Präsentations- und Schulungszwecke z.B. für neue Mitarbeiter eines Softwareunternehmens, zum anderen für die Identifikation von Optimierungsansätzen des Softwareentwicklungsprozesses mit dem letztendlichen Ziel, ein qualitativ hochwertiges Softwareprodukt zu erstellen. Unabhängig vom konkreten Ziel der Betrachtung startet diese zumeist mit der Erstellung von Prozessmodellen. Die Modelle sollen ein möglichst vollständiges und verständliches Bild der realen Prozesse widerspiegeln. Die Aufgabe der Erstellung solcher Modelle wird um einiges komplexer, wenn es um die Abbildung einer geografisch verteilten Entwicklung für Softwaresysteme geht.

Mit *Process Landscaping* wird im Rahmen dieser Dissertation eine Methode entwickelt und validiert, die bisher nicht oder nur unzureichend berücksichtigte Aspekte bei der Dokumentation und Analyse verteilt stattfindender Softwareentwicklungsprozesse beinhaltet. Dazu zählt insbesondere die Unterstützung bei der Erstellung eines konsistenten Gesamtbildes einer großen Menge von Prozessmodellen auf unterschiedlichen Abstraktionsebenen und dessen Analyse aus unterschiedlichen Blickwinkeln. Die unterschiedlichen Blickwinkel resultieren beim Process Landscaping aus einer logischen, die Durchführungsreihenfolge von Aktivitäten berücksichtigenden, und einer verteilungsspezifischen Sicht. Die Modellierung der oberen Abstraktionsebenen erlaubt insbesondere auch eine Analyse unstrukturierter Prozesse, die kein festes Ablaufverhalten aufweisen. Ein Schwerpunkt der Methode liegt auf der Identifikation, Dokumentation und Analyse von Schnittstellen zwischen Prozessen und zwischen einzelnen Aktivitäten innerhalb eines Prozesses. Dieser Schwerpunkt berücksichtigt damit auch den Verteilungsaspekt komplexer Softwareentwicklungsprozesse. Die Analyseziele des Process Landscaping konzentrieren sich auf verschiedene Kommunikationseigenschaften der betrachteten Prozesse, die sich entweder aus ihrer statischen Kommunikationsinfrastruktur ergeben oder aus ihrem (dynamischen) Kommunikationsverhalten. Für die Anwendung der Methode wird eine Werkzeugunterstützung zur Modellierung und Simulation präsentiert.

Danksagung

Diese Arbeit entstand am Lehrstuhl für Software-Technologie unter der Betreuung von Herrn Prof. Dr. Volker Gruhn als Erstgutachter und Herrn Prof. Dr. Ernst-Erich Doberkat als Zweitgutachter.

Bei der Erstellung dieser Dissertation wurde ich von vielen Personen unterstützt. Ohne ihre Hilfsbereitschaft wäre es mir unmöglich gewesen, die Arbeit in der vorliegenden Form zu erstellen.

Herrn Prof. Dr. Gruhn danke ich für die Betreuung während der Erstellung dieser Arbeit und für die Möglichkeit, meine Forschungsarbeit auch über Lehrveranstaltungen in Form von Projektgruppen und Seminaren vorantreiben zu können. Herrn Prof. Dr. Ernst-Erich Doberkat danke ich für die bereitwillige Übernahme des Zweitgutachtens. Die intensiven Gespräche mit ihm und seine detaillierten Anmerkungen waren mir eine wertvolle Hilfe.

Zunächst möchte ich mich bei allen Kollegen des Lehrstuhls für Software-Technologie bedanken. Die gute kollegiale Atmosphäre hat mir sehr geholfen. Mein besonderer Dank gilt hier Klaus Alfert, der mit mir in vielen Stunden konstruktiver Gespräche die formale Seite der Petrinetze diskutiert hat. Seine zahlreichen Anmerkungen gegen Ende der Arbeit waren in gleichem Maße von großem Wert. Georgios Lajios danke ich für seine Unterstützung beim kritischen Umgang mit selbst erstellten Definitionen.

Den Studenten, die meine Forschungsarbeiten durch Diplomarbeiten und Projektgruppen unterstützt haben, gebührt ebenfalls ein besonderer Dank. In alphabetischer Reihenfolge waren dies Carsten Brockmann, Andreas Pohlmann, Hartmut Reichelt, Marc Störzel sowie alle Teilnehmer der Projektgruppe Palermo.

Auch außerhalb des Lehrstuhls haben viele Personen zum Gelingen dieser Arbeit beigetragen. Hier möchte ich vor allem Joachim Coutelle und Jörg Dickmann danken. Ihre Geduld und Mühe, die sich durch viele Stunden hilfreichen Kritisierens, geduldigen Zuhörens und intensiver Diskussionen geäußert haben, haben zur Identifikation vieler Verbesserungsmöglichkeiten und damit zum Fortschreiten dieser Arbeit beigetragen. Ingrid Reck danke ich besonders herzlich für ihr intensives Korrekturlesen meiner Arbeit und für ihre ständige Bereitschaft zur Hilfestellung bei Formulierungsfragen.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Einführung in das Forschungsgebiet der Softwareprozesstechnologie

Seit mehr als 25 Jahren entstehen auf dem Forschungsgebiet der Softwareprozesstechnologie und in der Industrie immer neue Vorgehensmodelle zur Unterstützung von Softwareentwicklungsprozessen, die einem besseren Management dieser Prozesse dienen sollen. Unter einem Vorgehensmodell ist hier eine Darstellung gemeint, „die weitgehend den Softwareentwicklungsprozess beschreibt und auch Analysen des Prozesses gestattet“ [Chr92]. Die Darstellung komplexer Softwareentwicklungsprozesse erfordert nicht nur die Berücksichtigung der traditionellen Entwicklungsphasen wie Anforderungsdefinition, Analyse, Entwurf, Implementation und Test, sie verlangt eine ganzheitliche Betrachtung aller Aktivitäten rund um ein Softwareentwicklungsprojekt. Dazu gehört das Projektmanagement mit all seinen organisatorischen und strategischen Aufgaben ebenso wie das Qualitätsmanagement und das Konfigurationsmanagement. Mit einer solchen ganzheitlichen Betrachtung wird eine Reihe unterschiedlicher Ziele verfolgt. Dabei bildet die Dokumentation der Prozesse die Grundlage für Präsentations- und Schulungszwecke z.B. für neue Mitarbeiter eines Softwareunternehmens; ihre Analyse dient der Identifikation von Optimierungsansätzen des Softwareentwicklungsprozesses mit dem letztendlichen Ziel, ein qualitativ hochwertiges Softwareprodukt zu erstellen.

Dass der Trend zur Erarbeitung immer weiterer Vorgehensmodelle auch weiterhin nicht an Aktualität verliert, hat verschiedene Gründe:

- Entwicklungsparadigmen wie z.B. die objektorientierte oder die komponentenbasierte Softwareentwicklung haben einen großen Einfluss auf die Vorgehensstrategien und damit auch auf die Vorgehensmodelle. Mit der Evolution existierender und der Generierung neuer Paradigmen sind auch die Vorgehensmodelle einem stetigen Wandel unterworfen. Im Zusammenhang mit der objektorientierten Softwareentwicklung ist beispielsweise der *Unified Software Development Process* entwickelt worden [JBR98], im Rahmen der komponenten-

basierten Softwareentwicklung ist unter anderem *Select Perspective* entstanden [AF98].

- Viele Vorgehensmodelle sind für spezielle Anwendungsdomänen erstellt worden. Neue Domänen, wie zum Beispiel der e-business- und der Multimedia-Bereich, erfordern spezielle und damit auch wieder neue Vorgehensmodelle. Aktuell beschäftigt sich die Softwareprozesstechnologie verstärkt mit Unterstützungsmöglichkeiten für das Gebiet des Mobile Computing [WC01] und – Anwendungsgebiete übergreifend – Web-basierten Prozessen. Die unterstützenden Web-basierten Systeme bezeichnen in diesem Zusammenhang „Software, die es ermöglicht, Geschäftsvorfälle über das World Wide Web abzuwickeln“ [Col02].

- Die zunehmende Globalisierung vieler Unternehmen hat einen erhöhten Verteilungsgrad eines Softwareprozesses zur Folge, der wiederum andere Vorgehensweisen für sein Management notwendig macht. Die RWTH Aachen beschäftigt sich beispielsweise mit der Delegation verteilter Prozesse [BJSW01]. Aber auch die zuvor genannten Anwendungsdomänen zeichnen sich durch Prozesse aus, die insbesondere den Verteilungsaspekt beinhalten.

Neben diesen eher aspektorientierten Forschungsgebieten rund um das Thema der Vorgehensmodelle gibt es Bemühungen, verbesserte Beschreibungsmöglichkeiten für Prozesse zu entwickeln. Hier ist vor allem das Schlagwort *Process Pattern* zu nennen [Stö01a]. Die ersten Arbeiten zum Thema Pattern allgemein erschienen Ende der 70er Jahre unter anderem mit [AISJ77] und [Ale79]. Ein Pattern wird hier wie folgt beschrieben:

„Each pattern describes a problem that occurs over and over again in our environment and then describes the core of the solution to that problem in such a way that you can use this solution a million times over without ever doing it the same way twice.“ (zitiert in [RJ00]).

Das Konzept der Pattern-Verwendung bezieht sich dort auf das Anwendungsgebiet der Gebäudearchitektur. Es wurde von Gamma et al. mit der Entwicklung von Design Pattern auf das Gebiet der Softwaretechnologie übertragen [GHJV94] und findet inzwischen auch auf dem Gebiet der Softwareprozesstechnologie Verwendung [Amb98]. Der Einsatz von Process Pattern unterstützt jedoch keine dynamischen Analysen und auch nicht die Automatisierung der zugrundeliegenden Prozesse. Aus diesem Grund werden auch weiterhin bereits bekannte Prozessmodellierungssprachen wie unter anderem Little-Jil [SOSW98], APEL [DEA98] oder die ereignisgesteuerten Prozessketten [LSW97] sowie die vielfach eingesetzten und in unterschiedlichen Varianten existierenden Petrinetze [Bau96] verwendet und weiterentwickelt.

Ein weiteres Schlagwort auf dem Gebiet der Softwareprozesstechnologie ist *PSEE* (Process-sensitive Software Engineering Environment). Es ist im Zusammenhang mit einem der Kernziele der Softwareprozesstechnologie entstanden, der informationstechnischen Prozessunterstützung [DKW99]. Eine PSEE wird von einer *Process Engine*, beispielsweise einem Interpreter einer Prozessmodellierungssprache, gesteuert.

Diese kontrolliert den Informationsfluss zwischen den Arbeitsplätzen der verschiedenen Prozessbeteiligten gemäß einem zugrundeliegenden Prozessmodell. Letzteres wird zusammen mit Informationen über ein zu entwickelndes Softwareprodukt sowie den Prozessstatus in einem Repository verwaltet.

Unabhängig vom konkreten Ziel der Dokumentation, Analyse oder informationstechnischen Unterstützung verschiedener Prozesse erfordert jedes Ziel zunächst die Erstellung von Prozessmodellen. Sie dienen sowohl als fachlich bestimmte Grundlage der Softwarerealisierung als auch teilweise zur Festlegung der Kontrollstrukturen eines Prozesses. Die Modelle sollen ein möglichst vollständiges und verständliches Bild der realen Prozesse widerspiegeln. Die Aufgabe der Erstellung solcher Modelle wird um einiges komplexer, wenn es um die Abbildung einer geografisch verteilten Softwareentwicklung für Softwaresysteme geht. Die Berücksichtigung von Schnittstellen zwischen und innerhalb von Prozessen gewinnt hier besonders an Bedeutung.

In [CDP95] werden laut der Autoren erstmals Schnittstellen explizit betrachtet bzw. wird beschrieben, welche Erkenntnisgewinne dabei erzielt werden konnten. Die Vorgehensweise für diese Betrachtung war hierbei bottom-up, d.h. man identifizierte und beschrieb die Schnittstellen zwischen den Prozessen zweier Unternehmen entlang ihrer Prozessmodelle und fasste sie zu Gruppen zusammen. Auf diese Weise konnten neben der stark variierenden Granularität der Schnittstellenbeschreibungen auch viele Inkonsistenzen sowohl in den Prozessen als auch in den entsprechenden Prozessmodellen aufgedeckt werden. Ein Top-Down-Ansatz, der Schnittstellen von Beginn eines Modellierungsprojektes an explizit berücksichtigt, würde einer starken Varianz der Granularität entgegenwirken, Inkonsistenzen im Prozess eher aufdecken und im Modell gänzlich vermeiden, weil diese schon bei der Wissensakquise vor der Modellerstellung oder spätestens während der Modellierung identifiziert würden. Ein solcher Ansatz ist – zumindest in den einschlägigen Zeitschriften und auf Konferenzen, die sich mit dem Thema der Softwareprozessmodellierung beschäftigen – nicht identifiziert worden. Die vorangegangene Diskussion zeigt hier jedoch deutlich einen entsprechenden Handlungsbedarf.

Neben der Berücksichtigung von Schnittstellen bei der Modellierung von geografisch verteilt durchgeführter Softwareentwicklung spielt die Kommunikation der verteilt arbeitenden Prozessbeteiligten eine immer bedeutendere Rolle. Diese Kommunikation wird derzeit meist mit Fokus auf deren informationstechnische Unterstützung, etwa durch Workflowmanagementsysteme [BT98] oder andere CSCW-Systeme (Groupware, etc.) [Hay00], analysiert. Eine Kommunikationsanalyse mit dem Ziel der Kommunikationsoptimierung erfordert jedoch nicht nur eine Analyse der hierfür eingesetzten Systeme, sondern auch des Kommunikationsverhaltens und der systemunabhängigen Eigenschaften einer Kommunikationsinfrastruktur auf einem anderen Abstraktionsniveau. Dieses muss die geografische Verteilung der Kommunikationspartner und ihre Kommunikationsintensität untereinander stärker als bislang berücksichtigen. Sitzen Prozessbeteiligte, die häufig miteinander kommunizieren müssen, in einem verteilt ablaufenden Prozess geografisch weit auseinander, kann kein noch so performantes Kommunikationssystem diese ineffiziente Verteilung der partizipierenden Pro-

zessbeteiligten ausgleichen. Da die bislang zur Verfügung stehenden Analysemethoden das Kommunikationsverhalten der Prozessbeteiligten nicht unter dem Aspekt ihrer geografischen Verteilung untersuchen und die für die Kommunikation erforderlichen Eigenschaften einer zugrundeliegenden Infrastruktur immer systemabhängig betrachtet werden, ergeben sich neue Anforderungen an diese Analysemethoden bzw. es ergibt sich die Forderung nach geeigneten neuen.

Eine Konzentration auf Kommunikationseigenschaften von verteilten Prozessen bezüglich Infrastruktur und Verhalten erfordert für deren Analyse eine geeignete Modellierung, die auch die Fragen beantwortet, wo welche Informationen gehalten werden und welche Prozessteilnehmer diese über welche Standorte hinweg austauschen. Diese Fragen können teilweise, wie bei der Fokussierung auf andere Aspekte innerhalb eines Softwareentwicklungsprozesses (beispielsweise verschiedene beteiligte Rollen) oft propagiert, durch die Erstellung spezifischer Prozessmodelle beantwortet werden, deren Integration einen Gesamtüberblick über alle Aspekte der modellierten Prozesse verschafft [FKN+92]. Diese Integration ist jedoch mit einem großen Kontrollaufwand verbunden, da sowohl Redundanzen als auch Inkonsistenzen entstehen können. Wünschenswert wäre eine Möglichkeit der Fokussierung durch die Veränderung des Blickwinkels auf ein einmal erstelltes Prozessmodell, d.h. eine Umordnung der Modellelemente von der Betrachtung ihrer logischen Abfolge zu einer Betrachtung der Verteilung auf verschiedene Standorte. Die daraus resultierenden Erkenntnisse können möglicherweise Aufschluss über Umstände geben, die bei der bisherigen Modellierung nicht ohne weiteres erkennbar waren. Die anhaltende Diskussion über die Frage nach geeigneten Sichten zur Analyse bestimmter Aspekte und/oder der Integration bzw. Aggregation von Teilmodellen zu einem Gesamtmodell (vgl. beispielsweise [MENW99, GEMT00]) zeigt, dass auch hier noch immer Forschungsbedarf vorhanden ist.

Alle bislang angesprochenen Aspekte der (Software-)Prozessmodellierung werden hauptsächlich bei strukturierten Prozessen untersucht. Diese klar definierten und wiederholbaren Prozesse sind relativ einfach abzubilden und z.B. durch Workflowmanagementsysteme gut zu unterstützen. Es gibt jedoch insbesondere in Klein- und Mittelbetrieben eine Vielzahl von Softwareprozessen, die sowohl auf Kreativität basieren als auch auf daraus resultierender Flexibilität ihres Ablaufs. Diese aufgrund des Fehlens einer festen Ablaufreihenfolge auch als unstrukturiert bezeichneten Prozesse werden zurzeit nicht oder kaum durch geeignete Prozessmodellierung unterstützt. Oft behilft man sich mit der Modellierung aller möglichen Ablaufalternativen, was jedoch sehr aufwändig ist, immer die Gefahr der Unvollständigkeit birgt und die Kreativität der Prozessbeteiligten einschränkt. Zudem ist das Erkennen von Schwachstellen unstrukturierter Prozesse und/oder eine gezielte Modifikation ohne eine angemessene Dokumentation nicht oder nur schwer möglich. Für das Messen der Auswirkungen von Modifikationen gilt das gleiche; weitere Gründe sind in [DD98] aufgeführt. Die Suche nach einer geeigneten Form der Modellierung für diese Art von Softwareprozessen ist bislang noch nicht abgeschlossen und erfordert daher ebenfalls das Ergreifen entsprechender Maßnahmen.

Manchmal ist es wünschenswert, Prozesse zunächst auf einem hohen Abstraktionsniveau abzubilden und sie erst zu einem späteren Zeitpunkt für bestimmte Analysen zu verfeinern. Für diese Situationen ist es hilfreich, eine geeignete Vorgehensweise für die Modellierung zur Hand zu haben, die über entsprechende Verfeinerungsmechanismen verfügt, so dass eine Neumodellierung für detaillierte Abbildungen entfällt und so die Gefahr von Inkonsistenzen zwischen den groben und den detaillierten Modellen gar nicht erst aufkommt. Zur Unterstützung von Prozessbeteiligten empfiehlt sich zusätzlich eine geeignete grafische Darstellung der Prozessmodelle, da diese meist leichter verständlich sind als mathematische Modelle. Für genaue Analysen ist jedoch eine formale Grundlage erforderlich. Petrinetze erfüllen diese Anforderungen: Über die Möglichkeit von Transitionsverfeinerungen ist ein geeignetes Verfeinerungskonzept zur Erstellung verschiedener Abstraktionsebenen gegeben [Bau96]. Die grafische Darstellung eines Petrinetzes als bipartiter Graph erlaubt eine Visualisierung der Prozessmodelle, während seine formale Grundlage exakte Analysen ermöglicht.

Die bekannten Petrinetz-Varianten haben jedoch den Nachteil, dass bei der Modellierung immer auch ein Ablauf berücksichtigt werden muss, um das Modell formal korrekt zu halten. Dies ist aber gerade bei den zuvor erwähnten unstrukturierten Prozessen oft nicht gewünscht. Für sie ist daher eine geeignete Darstellungsform zu entwickeln.

Zusammenfassend sind in diesem Abschnitt fünf verschiedene Problemstellungen im Bereich der Softwareprozesstechnologie identifiziert worden, nämlich

1. die mangelnde explizite Berücksichtigung von Schnittstellen innerhalb von und zwischen Softwareprozessen,
2. die mangelnde Berücksichtigung der geografischen Verteilung von Prozessbeteiligten und die daraus resultierenden Auswirkungen auf deren Kommunikationseffizienz,
3. die fehlende Möglichkeit der Betrachtung von Softwareprozessmodellen unter verschiedenen Blickwinkeln ohne den Aufwand einer Neumodellierung oder einer Integration verschiedener Sichten,
4. die mangelnde Modellierungsunterstützung für unstrukturierte Softwareprozesse und schließlich
5. die mangelnde durchgängige Modellierungsunterstützung für Softwareprozesse, die sowohl auf sehr abstraktem als auch – zumindest in Teilbereichen – auf sehr detailliertem Niveau betrachtet werden sollen und die neben einer mathematisch präzisen Darstellung auch ihre Visualisierung ermöglicht.

Aus diesen identifizierten und hier aufgeführten Problemstellungen ist die Motivation für die vorliegende Arbeit entstanden. Die Erarbeitung geeigneter Lösungsmöglichkeiten verbessert die Unterstützung sowohl der Prozessbeteiligten als auch des Prozessmanagements, deren gemeinsames Ziel es ist, mit Hilfe eines verbesserten Softwareprozesses effektiver ein qualitativ hochwertiges Softwareprodukt zu erstellen. Die

mangelnde Berücksichtigung von Schnittstellen zwischen Prozessen birgt insbesondere durch die aktuell zunehmende Globalisierung von Unternehmen und der damit einhergehenden geografischen Verteilung von Prozessen eine wachsende Gefahr, die Menge der an einer Softwareentwicklung beteiligten Teilprozesse nur punktuell zu verbessern, ohne eine tatsächliche Optimierung des Gesamtprozesses zu erreichen. Die geografische Verteilung der Prozessbeteiligten erfordert des Weiteren eine Analyse und ggf. Optimierung des Kommunikationsverhaltens und systemunabhängiger Kommunikationsstrukturen, die – wie bereits diskutiert – nicht durch die Analyse und Optimierung verwendeter Kommunikationssysteme ersetzt werden können. Die Umstrukturierung von Softwareprozessmodellen zur Erstellung verschiedener Sichten vermeidet mögliche Inkonsistenzen in den Modellen, die bislang nur unzureichend gehandhabt werden können. Die fehlende oder zumindest unzureichende Modellierungsunterstützung für unstrukturierte Softwareprozesse nimmt Klein- und Mittelbetrieben bislang die Möglichkeit, durch verbesserte Softwareprozesse eine Qualitätssteigerung ihrer Softwareprodukte zu erreichen. Und schließlich werden auch viele Softwareunternehmen durch die mangelnde durchgängige Modellierung ihrer Prozesse auf verschiedenen Abstraktionsebenen davon abgehalten, die Prozessmodellierung als Mittel zur Identifikation von Optimierungsansätzen einzusetzen.

Die Diskussion der Problemstellungen zeigt die Notwendigkeit auf, entsprechende Lösungsansätze zu erarbeiten und diese den Prozessbeteiligten und -verantwortlichen zur Verfügung zu stellen. Die Erarbeitung dieser Lösungsansätze ist Inhalt der vorliegenden Arbeit. Ihre konkrete Zielsetzung wird im folgenden Abschnitt erläutert.

1.2 Zielsetzung der Arbeit

Im Rahmen dieser Dissertation wird eine Methode entwickelt und validiert, mit der verteilte Softwareprozesse modelliert und analysiert werden können. Ziel der Methode ist es, die im vorangegangenen Abschnitt 1.1 aufgeführten Problemstellungen aufzugreifen und geeignete Lösungsmöglichkeiten anzubieten. Bevor die Methode in Kapitel 2 im Detail vorgestellt wird, wird kurz erläutert, wie sie zur Lösung dieser Problemstellungen beiträgt.

Die Modellierung von Prozessmodellen setzt hier auf eine strukturierte und durchgängige Vorgehensweise, mit der nach dem Top-Down-Ansatz eine zusammenhängende Menge von Prozessen über verschiedene Abstraktionsebenen mit unterschiedlichem Detaillierungsgrad abgebildet wird, ohne dass dabei der Gesamtüberblick verloren geht. Dazu werden dem Anwender für jeden Modellierungsschritt Techniken zur Wissensakquise angeboten, die für den jeweils gewünschten Detaillierungsgrad angemessen sind. Die durch die Darstellung der Softwareprozesse entstehende Menge von Prozessmodellen, nachfolgend als Prozesslandschaft bezeichnet, hat der Methode ihren Namen gegeben:

Process Landscaping

Ein Schwerpunkt des Process Landscaping liegt auf der Identifikation, Dokumentation und Analyse von prozessübergreifenden und prozessinternen Schnittstellen zwischen Aktivitäten innerhalb dieser Prozessmodelle. Die Methode berücksichtigt explizit die im vorangegangenen Abschnitt zuerst genannte Problemstellung, indem bereits auf der obersten Abstraktionsebene eines jeden Modells zumindest die Existenz von Schnittstellen abgebildet wird. Diese Information kann über mehrere Verfeinerungsschritte nach und nach detailliert werden, bis für eine Schnittstelle eine umfassende Beschreibung bezüglich ihres Inhalts, der auf sie zugreifenden Prozessbeteiligten, etc. vorliegt.

Verbinden die Schnittstellen gleichzeitig verschiedene Standorte, an denen Prozesse ausgeführt werden, wird insbesondere auch der Verteilungsaspekt komplexer Softwareprozesse (und damit die zweite Problemstellung) berücksichtigt. Über diese geografische Verteilung können Anforderungen an die zugrundeliegende Kommunikationsinfrastruktur gestellt und überprüft werden. Auch das Kommunikationsverhalten innerhalb eines verteilten Softwareprozesses und seine damit zusammenhängende Wirtschaftlichkeit können über die Schnittstellenbetrachtung zwischen verschiedenen Standorten analysiert werden.

Überprüfung und Analyse verschiedener Kommunikationsaspekte werden durch eine auf die entsprechenden Modelleigenschaften konzentrierte Sicht erleichtert, die durch die Umstrukturierung einer bereits existierenden Prozesslandschaft entsteht. Diese ist durch die traditionelle Vorgehensweise der Prozessmodellierung entstanden, die ein Modell entlang der kausalen Zusammenhänge verschiedener durchzuführender Aufgaben innerhalb des abzubildenden Prozesses entwirft. Ihre Umstrukturierung erfüllt die Anforderung an die Existenz verschiedener Sichten, die durch die dritte angesprochene Problemstellung aufgestellt worden ist, konkret der Anforderung nach dem Verzicht auf Neumodellierung und/oder Integration verschiedener Sichten.

Der Top-Down-Ansatz des Process Landscaping ermöglicht die Berücksichtigung der Problemstellungen 1. bis 3. nicht nur für strukturierte, sondern auch für unstrukturierte Softwareprozesse. Dazu wird eine Vorgehensweise der Prozessmodellierung eingeführt, die zunächst explizit von Ablaufinformationen abstrahiert und so gleichzeitig die in der vierten Problemstellung formulierte Anforderung erfüllt. Verfeinerungskonzepte, die sowohl das spätere Hinzufügen der Ablaufinformationen als auch eine schrittweise Detaillierung verschiedener Prozesselemente erlauben, sichern schließlich eine durchgängige Modellierungsunterstützung für Softwareprozesse, wie sie im Rahmen der fünften Problemstellung diskutiert worden ist.

Bezüglich der Analyseziele des Process Landscaping sind noch einige Charakteristika hervorzuheben. Die Methode verfolgt eine prozessorientierte Analyse, d.h. der Fokus liegt nicht auf der Qualität der Ergebnisse eines Softwareprozesses, sondern auf der Qualität des Prozesses selbst. Der Vorteil dieses Analyseansatzes liegt darin, dass Schwächen des Prozesses bereits identifiziert werden können, bevor das Modell ausgeführt wird, indem verschiedene Modellvarianten, die das gleiche Ziel (beispielsweise Aufbau eines weiteren Standortes im Rahmen eines komponentenbasierten Softwareentwicklungsprojektes) haben, analysiert und verglichen werden können. Durch die Verbesserung von Prozessmodellen kann so schließlich auch die Qualität der Soft-

wareprodukte verbessert werden, die entlang dieser Prozessmodelle erstellt werden.

Im Rahmen der Analyse des Process Landscaping werden verschiedene Eigenschaften einer Softwareprozesslandschaft untersucht. Dies sind zum einen diejenigen Eigenschaften statischer Kommunikationsstrukturen, die Auswirkungen auf die Kommunikationskosten verteilter Prozesse haben, zum anderen sind dies Eigenschaften, die die Wirtschaftlichkeit des Kommunikationsverhaltens, welches sich aus der Prozessdynamik ergibt, beeinflussen. Für die Analyse wird jedoch aufgrund ihrer Individualität keine allgemeingültige Metrik zur Qualitätsmessung der analysierten Landschaften angegeben. Es wird stattdessen mit Hilfe von Vergleichsanalysen und unter der Annahme konkreter Planungsideen das jeweilige Kostenoptimum für eine gegebene Prozesslandschaft berechnet und so quantitativ nachvollziehbar das Ziel einer qualitativ hochwertigen Prozesslandschaft angestrebt.

Für die Anwendung des Process Landscaping ist eine Werkzeugunterstützung zur Modellierung und Simulation – als eine Form der Analyse – implementiert. Methode und Werkzeug werden an einem durchgängigen Beispiel erläutert und validiert. Voraussetzung für dieses Vorhaben ist neben der exakten Definition verwendeter Begriffe und der Erläuterung der Vorgehensweise beim Einsatz des Process Landscaping die Erarbeitung einer formalen Grundlage. Die wichtigsten im Rahmen des Process Landscaping verwendeten Begriffe werden im nachfolgenden Abschnitt eingeführt.

1.3 Terminologie

Innerhalb dieses Abschnitts wird eine Grundmenge von Begriffen definiert, die in der Arbeit verwendet wird. Diese Menge erhebt nicht den Anspruch, ein umfassendes Glossar aller im Bereich der Softwareprozesstechnologie verwendeten Begriffe zu repräsentieren. Sie deckt lediglich eine Kernmenge der im Rahmen dieser Arbeit benutzten Begriffe rund um Softwareprozesse und -prozessmodelle ab.

Bei der Methode des Process Landscaping steht die Modellierung einer Menge von zusammenhängenden Prozessen im Zentrum der Betrachtungen. Sowohl die Prozesse als auch ihr Zusammenhang können dabei unterschiedlich detailliert modelliert werden. Eine *Prozesslandschaft* ist eine Abbildung von realen Prozessen in Form eines Modells auf unterschiedlichen Abstraktionsebenen. Abbildung 1.1 verdeutlicht den Zusammenhang zwischen realen Prozessen und ihrer Darstellung innerhalb einer Prozesslandschaft.

Die Elemente der Prozesslandschaft sind rechts benannt. Diese repräsentieren Ausschnitte einer realen Welt, deren korrespondierende Elemente auf der linken Seite der Abbildung aufgeführt sind. Die Relationen zwischen Elementen der realen Welt sind – genauso wie die Relationen zwischen den sie darstellenden Elementen der Prozesslandschaft – durch das Symbol einer Aggregation veranschaulicht. Die Korrespondenz zwischen den Elementen der realen Welt und der Prozesslandschaft wird durch das Symbol der Assoziation angezeigt.

Auf der obersten Abstraktionsebene einer Prozesslandschaft werden *Kernaktivitätsbe-*

schreibungen (kurz *Kernaktivitäten*) abgebildet, die die *Kernprozesse* eines Softwareprojektes darstellen. Jeder Kernprozess besteht aus einer Menge von *Prozessen*, die wiederum in *Teilprozesse* und *Aktivitäten* aufgeteilt sind. Innerhalb der Modellwelt, der Prozesslandschaft, werden diese Bestandteile der Prozesse als *Aktivitätsbeschreibungen* (kurz *Aktivitäten*) oder als *Prozessmodelle* bezeichnet, die im Modell durch Verfeinerung der Kernaktivitäten und Aktivitäten entstehen. Der Unterschied zwischen den beiden Modellbegriffen *Aktivität* und *Prozessmodell* besteht darin, dass ein Prozessmodell explizit von der Betrachtung der Ablaufinformationen ausgeht, die die Teile der Prozesse miteinander verbinden. Spricht man dagegen in der Modellwelt von einer Aktivität, müssen diese Ablaufinformationen nicht notwendigerweise abgebildet sein. Der Modellbegriff *Aktivität* ist damit als Oberbegriff für abgebildete Prozesse, Teilprozesse oder Aktivitäten anzusehen.

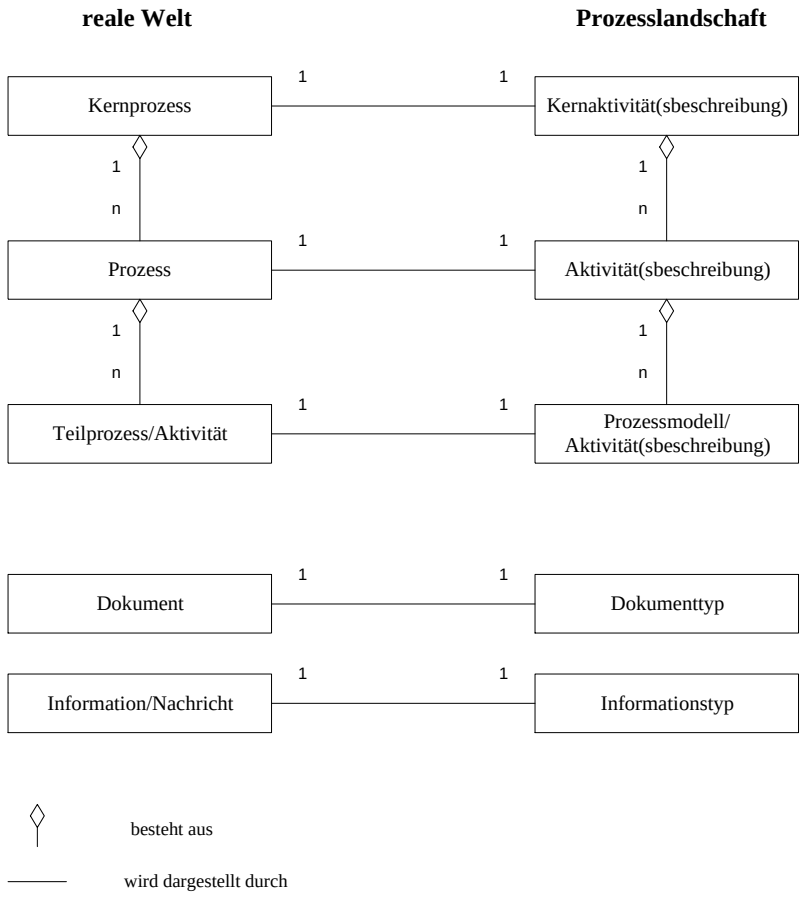


Abbildung 1.1: Prozesse, Aktivitäten und ihre Darstellung in einer Prozesslandschaft

Aufgrund der Unterscheidung zwischen Aktivitäten mit und ohne Ablaufinformationen wird ihm Rahmen des Process Landscaping bewusst von *Verfeinerung bis auf*

Prozessmodellebene gesprochen (vgl. Abschnitt 2.2.3). Die Ablaufinformationen von Prozessen unterteilen damit eine Prozesslandschaft in eine Anzahl *oberer* und *unterer Ebenen*. Auf ihren oberen Ebenen werden diese explizit nicht abgebildet; auf den unteren Ebenen fließen sie in die Darstellung mit ein.

Informationsobjekte werden innerhalb einer Prozesslandschaft ebenfalls in unterschiedlichen Detaillierungsstufen abgebildet. Auf den oberen Ebenen sind ausschließlich Dokumente abgebildet, die sich dort als Dokumenttypen wiederfinden. Ein *Dokument* ist eine Menge von Informationen oder Materialien, die im Rahmen eines Prozesses erzeugt oder bearbeitet werden. Auf den unteren Ebenen kann eine Vielzahl unterschiedlicher Informationsobjekte abgebildet werden. Diese werden in der Modellwelt entsprechend als Informationstypen unterschiedlicher Art dargestellt, die meist als Nachrichten zwischen Aktivitäten ausgetauscht werden.

Die eingeführten Begriffe sind teilweise direkt aus der Literatur übernommen, teilweise unterscheidet sich ihre Bedeutung im Rahmen des Process Landscaping von diesen. Beispielsweise definieren Hammer und Champy 1993: „We define a process as a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer“ [HC93]. Diese Definition akzentuiert die Kundenorientierung, die auch beim Process Landscaping eine wichtige Rolle spielt (vgl. Abschnitt 2.2.2).

Beim Process Landscaping wird unter einem Prozess ein Softwareprozess eines Unternehmens verstanden, der als eine spezielle Form eines Geschäftsprozesses angesehen wird, konkret als diejenige, deren Ergebnis ein Softwareprodukt für einen tatsächlichen oder einen möglichen Kunden ist. Dies entspricht dem Verständnis von Keen, der sich bei der Zielgruppe eines Geschäftsprozesses auf Kunden eines Unternehmens oder einen Markt beschränkt [Kee97]. Er beschreibt einen Geschäftsprozess als eine Menge von Aktivitäten, die Informationen verarbeiten und ein Ergebnis produzieren. Dabei wurde die Menge von Aktivitäten dazu entworfen, ein bestimmtes Ergebnis für einen Kundenbereich oder einen Markt zu erzeugen.

Lonchamp definiert einen Softwareprozess als „a set of partially ordered process steps, with sets of related artifacts, human and computerized resources, organizational structures and constraints, intended to produce and maintain the requested software deliverables“ [Lon93]. Die partielle Ordnung der Prozessschritte impliziert – zumindest für Teilbereiche – Wissen über den Ablauf eines betrachteten Prozesses. Damit betont Lonchamp im Gegensatz zu Keen neben den beteiligten Ressourcen auch die Berücksichtigung von Ablaufinformationen. Für Geschäftsprozesse allgemein wird dies in ähnlicher Form von Becker und Vossen formuliert, die diese als „inhaltlich abgeschlossene zeitliche und sachlogische Abfolge der Funktionen, die zur Bearbeitung eines betriebswirtschaftlich relevanten Objektes notwendig ist“, ansehen [VB96]. Zusammenfassend wird durch die Begriffsdefinitionen von Lonchamp und Becker/Vossen das Vorhandensein eines strukturierten Prozessablaufs und dessen Berücksichtigung bei der Prozessmodellierung stets impliziert. Beim Process Landscaping ist die Berücksichtigung von Ablaufinformationen dagegen nicht zwangsläufig für alle Abstraktionsebenen einer Prozesslandschaft gefordert. Ihre oberen Ebenen abstrahieren sogar bewusst davon. Das Process Landscaping folgt damit nicht den beiden in der Lite-

ratur formulierten Definitionen. Warum die Möglichkeit der Abstraktion von Ablaufinformationen – insbesondere für Softwareprozesse – sinnvoll ist, wurde bereits in Abschnitt 1.2 erläutert.

Bezogen auf die Verwendung der Begriffe innerhalb der Modellwelt, die in Abbildung 1.1 aufgeführt sind, gibt es noch weitere Unterschiede zur Literatur. So wird eine Aktivität von Lonchamp als elementarer Prozessschritt bezeichnet, wobei „at the level of abstraction of the process description, an activity has no visible substructure“ [Lon93, FH93]. Diese Beschreibung einer Aktivität ist jedoch nicht für das Process Landscaping gültig, da sie einer Lösung der in Abschnitt 1.2 identifizierten Problemstellung der durchgängigen Modellierungsunterstützung auf unterschiedlichen Abstraktionsebenen inklusive ihrer Visualisierung hinderlich wäre. Bei der Methode kann eine Aktivität sehr wohl durch weitere (Teil-)Aktivitäten verfeinert werden. Das Konzept der Aktivitätsverfeinerung inklusive einer entsprechenden grafischen Darstellung ist elementarer Bestandteil des Process Landscaping und wird in den Abschnitten 2.2.3 und 2.2.5 detailliert diskutiert.

Ein Prozessmodell wird von Humphrey und Feiler definiert als eine abstrakte Repräsentation eines Prozesses auf Architektur-, Design- oder Definitionsebene [FH93]. Dieses kann mehr oder weniger formal sein und eine bestimmte Sicht auf einen Prozess widerspiegeln [Lon93]. Beim Process Landscaping wird diese Bedeutung noch um die explizite Berücksichtigung von Ablaufinformationen erweitert. Der Grund für die Erweiterung ist die unterschiedliche Verwendung des Begriffs *Prozessmodell*, der beim Process Landscaping einen Teilbereich innerhalb einer Prozesslandschaft beschreibt, während er bei Humphrey und Feiler als Oberbegriff für abgebildete Prozesse oder Teilprozesse verwendet wird. Diese Rolle des Oberbegriffs übernimmt beim Process Landscaping der Begriff der *Aktivität* (vgl. Beschreibung der Abbildung 1.1).

Zusätzlich erfährt der Begriff der *Sicht* auf eine Prozesslandschaft im Rahmen dieser Arbeit eine besondere Bedeutung. Lonchamp definiert diesen als „the particular approach of a software process conveyed by a process model“ [Lon93]. Er ist der Meinung, dass ein vollständiges Softwareprozessmodell oft die Integration verschiedener Modelle erfordert, die verschiedene Aspekte des Softwareprozesses, beispielsweise Management-, Qualitäts- und Entwicklungsaspekte, betrachten. Derniame et al. formulieren dies ähnlich. Sie unterscheiden Submodelle wie das Aktivitätenmodell, das Produktmodell, das Ressourcenmodell und das Rollenmodell, die verschiedene Aspekte derselben Prozesslandschaft darstellen [DKW99]. Finkelstein et al. haben ein Rahmenwerk zur Integration dieser verschiedenen Sichten entwickelt [FKN+92], welches allgemein anerkannt und vielfach genutzt wird [GMT99, FS96].

Die Verwendung des Begriffes *Sicht* im Rahmen des Process Landscaping entspricht der zitierten Definition von Lonchamp, wenn die Erstellung verschiedener Sichten zur Analyse verschiedener Aspekte als sinnvoll erachtet wird. Die Integration separat erstellter Modelle, die jeweils verschiedene Sichten reflektieren – ist jedoch explizit nicht gewünscht. Stattdessen wird eine Vorgehensweise zur Erstellung verschiedener Sichten gefordert, die – ausgehend von einer Sicht – durch Umstrukturierung der Modellelemente weitere Sichten entstehen lässt. Diese Vorgehensweise lässt die Nutzung

eines Rahmenwerks zur Integration der Sichten überflüssig werden.

Auch in der Wirtschaftsinformatik wird der Begriff der Sicht auf ein Prozessmodell verwendet: „Ein Prozessmodell stellt immer eine (spezielle) Sicht auf ein Unternehmen dar, indem es den Durchlauf eines Objektes fokussiert. Die Vielzahl an betriebswirtschaftlichen Objekten (z.B. Bestellungen, Aufträge, Rechnungen, Mahnungen, Baugruppen, Teile inklusive ihrer jeweiligen Spezialisierungen) schlägt sich bei einer umfangreichen Modellierung in einer entsprechenden Vielzahl an Prozessmodellen nieder." [Ros96]. Da hier die zumeist separat erstellten Prozessmodelle oft nicht näher zueinander in Beziehung gesetzt werden, droht die Gefahr von Einzelprozessanalysen, d.h. der Optimierung eines einzelnen Prozesses ohne Berücksichtigung von etwaigen Wechselwirkungen. Im Rahmen des Process Landscaping wird dieser Gefahr entgegenwirkt, indem nicht verschiedene Prozessmodelle separat entwickelt werden, die dann miteinander abzugleichen oder zu integrieren sind, sondern indem zunächst eine Prozesslandschaft erstellt wird, die später zur Konzentration auf bestimmte zu betrachtende Eigenschaften umstrukturiert wird. Das Process Landscaping startet dabei mit der Erstellung einer Prozesslandschaft, bei der der kausale Zusammenhang der einzelnen Aktivitäten im Vordergrund steht. Die entstehende hierarchisch aufgebaute Struktur wird als *logische Sicht* auf eine Prozesslandschaft bezeichnet. Soll für eine Analyse der Verteilungsaspekt dieser Landschaft stärker hervorgehoben werden, so wird sie in eine *lokale Sicht* umstrukturiert (vgl. Abschnitt 2.3.1). Diese gruppiert alle Aktivitäten einer Prozesslandschaft nach den Standorten, an denen diese durchgeführt werden. Die Bedeutung des Begriffes der lokalen Sicht unterscheidet sich hier jedoch von derjenigen in anderen Arbeiten im Kontext verteilter Prozesse. In [FKT00] bezeichnet beispielsweise die lokale Sicht einen Teil einer Prozesslandschaft, der an einem konkreten, einzelnen Standort ausgeführt wird. Teilweise wird auch der Begriff der verteilten Sicht für lokal (über verschiedene Standorte) verteilte Systeme verwendet [TE00].

In der lokalen Sicht werden im Rahmen des Process Landscaping verschiedene Kommunikationsaspekte analysiert. Unter dem Begriff der *Kommunikation* wird hier der zielgerichtete Austausch von Informationen zwischen zwei oder mehreren Aktivitäten innerhalb einer Prozesslandschaft verstanden, die an verschiedenen Standorten ausgeführt werden können. Die Informationen liegen als Dokumente vor, die mit Hilfe einer Kommunikationsinfrastruktur transportiert werden. Kommunikationsanalysen in der Informatik allgemein betrachten meist andere Aspekte, wie beispielsweise die einer Kommunikation zugrundeliegenden Protokolle verteilter Rechnersysteme [Kru84, HK00] oder die Architektur einer verteilten Anwendung, deren Komponenten miteinander kommunizieren bzw. Nachrichten austauschen [Emm00, GT00]. Damit unterscheidet sich die Verwendung des Begriffs beim Process Landscaping insofern von der üblichen, als hier eher technische Aspekte der Kommunikation außer acht gelassen werden und stattdessen bislang kaum betrachtete Aspekte, die sich aus der geografischen Verteilung der Prozessbeteiligten ergeben, in den Vordergrund der Betrachtung treten.

1.4 Struktur der Arbeit

Nach der Einführung in das Forschungsgebiet der Softwareprozesstechnologie, in das sich die vorliegende Arbeit einfügt, der Identifikation weiterer Unterstützungsmöglichkeiten verteilter Softwareprozesse, die mit dem Process Landscaping angeboten werden, der Diskussion der sich daraus ergebenden Zielsetzung dieser Arbeit und der Einführung der in diesem Zusammenhang verwendeten Begriffe sind die Voraussetzungen zur ausführlichen Erläuterung der Methode gegeben. Kapitel 2 führt zunächst die Ziele und Grundsätze sowie die methodischen Schritte des Process Landscaping ein, die zur Erreichung der festgelegten Ziele durchzuführen sind. Letztere decken sowohl Modellierungs- als auch Analyseziele ab. Ein abschließender Abschnitt beinhaltet einen Vergleich mit anderen existierenden Modellierungs- und Analyseansätzen (Abschnitt 2.4).

Eine formale Basis der Methode, die insbesondere für die Erreichung der Analyseziele erforderlich ist, wird ausführlich in Kapitel 3 diskutiert. Die dort eingeführten Definitionen fügen sich in die bekannte Petrinetz-Notation ein. Sie werden an einem durchgängigen Beispiel erläutert und motiviert. Eine vergleichende Diskussion mit anderen Petrinetz-Varianten wird abschließend in Abschnitt 3.3 durchgeführt.

Das zur Erläuterung der formalen Basis herangezogene Beispiel, die Darstellung einer komponentenbasierten, verteilten Softwareentwicklung, wird in Kapitel 4 auch zur Validation der Analysemethoden des Process Landscaping herangezogen. Zusammen mit der Methode des Process Landscaping sind einige unterstützende Softwarewerkzeuge entstanden und existierende Werkzeuge erweitert worden, die die Anwendung der Methode erleichtern. Kapitel 5 stellt deren Kernfunktionalitäten vor und zeigt auf, welche Bereiche der Methode derzeit eine Werkzeugunterstützung vorweisen können. Kapitel 6 schließt die Arbeit mit einer Zusammenfassung und einem Ausblick auf mögliche weiterführende Arbeiten ab.

Kapitel 2

Die Methode des Process Landscaping

In der Literatur wird unter dem Begriff *Methode* eine „planmäßige und begründete Vorgehensweise zur Erreichung festgelegter Ziele (i.a. im Rahmen festgelegter Prinzipien)“ verstanden. „Zu einer Methode gehören eine Notation, systematische Handlungsanweisungen und Regeln zur Überprüfung der Ergebnisse.“ [HMF92]. Diese Definition einer Methode ist im Informatik-Begriffsnetz der Gesellschaft für Informatik (GI) festgehalten [GF01]. Ihr liegen vorangegangene Definitionen verschiedener dort referenzierter Autoren zugrunde [HMF92, Bal82, Chr92]. Balzert sieht beispielsweise eine Methode als eine Handlungsvorschrift an, die beschreibt, wie – ausgehend von gegebenen Bedingungen – ein Ziel mit einer festgelegten Schrittfolge erreicht wird [Bal82]. Chroust verwendet synonym den Begriff *Methodik* [Chr92].

Die Methodendefinition der GI berücksichtigt alle Aspekte vorangegangener Definitionen und führt zusätzlich die Bestandteile einer Methode explizit auf, während ältere Definitionen hauptsächlich die Handlungsvorschriften einer Methode betrachten. Das Process Landscaping basiert auf der Methodendefinition der GI, da vor allem eine Notation als Bestandteil einer Methode unverzichtbar ist, wenn exakte Analysen durchgeführt werden sollen.

Dieses Kapitel ist entlang der laut Definition notwendigen Bestandteile einer Methode strukturiert, konkret den methodischen Schritten und den zu erreichenden Zielen bzw. Ergebnissen. Zunächst wird – nach einer Erläuterung zu beachtender Grundsätze – die Vorgehensweise in Form einzelner Schritte zur Erstellung und Überprüfung einer Prozesslandschaft als ein Ergebnis des Process Landscaping diskutiert. Anschließend werden die Analyseziele der Methode definiert, ihre Vorgehensweise und Interpretationsrahmen für eine quantitative Bewertung vorgestellt. Die zugrundeliegende formale Notation zur Beschreibung einer Prozesslandschaft und ihrer zu analysierenden Eigenschaften wird zunächst nicht betrachtet. Sie wird im nachfolgenden Kapitel 3 diskutiert.

2.1 Ziele und Grundsätze des Process Landscaping

Ein erstes Ziel des Process Landscaping ist die Erstellung einer konsistenten Softwareprozesslandschaft. Das Process Landscaping geht dabei von der Annahme aus, dass im Rahmen von Modellierungsprojekten unabhängig von ihren konkreten Zielen insbesondere folgende Fragestellungen diskutiert werden müssen [GW00a]:

- Welches sind die Kernprozesse im Rahmen einer Softwareentwicklung?
- In welcher Reihenfolge sollten sie modelliert werden?
- Wie sehen die Schnittstellen zwischen ihnen aus?

Alle drei Fragen beschäftigen sich indirekt mit der Gewährleistung der Konsistenz einer Prozesslandschaft: Sind alle Kernprozesse identifiziert, können auch alle Schnittstellen gefunden werden. Deren Identifikation wird zusätzlich durch eine geeignete Modellierungsreihenfolge unterstützt. Sind die Schnittstellen beschrieben, können sie auf ihre Konsistenz überprüft werden. Dies alles führt letztendlich zu einer konsistenten Softwareprozesslandschaft.

Erst die Beantwortung der obigen Fragen ermöglicht eine zusammenhängende Betrachtung aller beteiligten Prozesse. Fehlt diese Möglichkeit, entsteht leicht eine Vielzahl einzelner Modelle, deren Zusammenhang nicht erkennbar ist. Eine Fokussierung auf Details erschwert insbesondere eine stringente Modellierung und Modifizierung von Schnittstellen. Ein zusammenhängendes Gesamtbild ermöglicht daher die Identifikation von Verbesserungspotenzial der modellierten Prozesse. Aufgrund dieser Erkenntnis stehen bei der Methode des Process Landscaping nicht einzelne Prozesse, sondern die Menge der zusammenhängenden Prozesse – deren Modell kurz als Prozesslandschaft bezeichnet wird – im Zentrum der Betrachtungen.

Der Modellierung einer Prozesslandschaft mit der Methode des Process Landscaping sind verschiedene Prinzipien zugrundegelegt:

- Anzahl von Kernprozessen:
  Das Process Landscaping geht von einer kleinen Anzahl zu identifizierender und zu betrachtender Kernprozesse aus [GW99a]. Sie soll die Übersichtlichkeit der obersten Ebene einer Softwareprozesslandschaft gewährleisten.

- Modellierungsreihenfolge von Prozessen:
  Das Process Landscaping geht von der Annahme aus, dass ein Gesamtüberblick über die Menge aller an einer Softwareentwicklung beteiligten Prozesse für die Erstellung einer Prozesslandschaft nutzbringend ist. Ein erster Überblick wird durch die Identifikation und Auflistung aller Kernprozesse erreicht. Die vollständige Identifikation dieser Kernprozesse kann dadurch gewährleistet werden, dass Kernprozesse in der Reihenfolge ihrer Nutzung in einem Kundenlebenszyklus vom Modellierer – meist im Rahmen eines Interviews – erfragt werden.

Unter einem Kundenlebenszyklus wird hier der Lebenslauf eines Softwaresystems von Projektbeginn an über Nutzung und Betreuung bis zur Außerbetriebnahme – also ein Softwarelebenszyklus [GF01] – verstanden, der aus der Sicht eines potentiellen Kunden betrachtet wird. Das Process Landscaping setzt dabei voraus, dass – meist zu Beginn eines Softwareentwicklungsprojektes – mindestens einer der Kernprozesse eine Schnittstelle zu einem potentiellen Kunden hat. Der Begriff des Kunden wird hierbei sehr weit gefasst, da es sich bei ihm um keinen konkreten Auftraggeber oder Anwender handeln muss. Die Anlehnung der Modellierungsreihenfolge an einen Kundenlebenszyklus unterstützt ein zielgerichtetes Vorgehen und gewährleistet die Berücksichtigung aller beteiligten Kernprozesse.

- Kernprozesse mit Kundenschnittstellen:
  Das Process Landscaping unterscheidet zwischen Kernprozessen mit direkten Schnittstellen zu Kunden und internen, unterstützenden (Kern-)Prozessen. Diese Unterscheidung folgt der Empfehlung von Höderath [Höd98], der zwischen Marktprozessen und internen Prozessen unterscheidet. Bei ersteren existiert eine direkte Kommunikation mit Kunden und damit direkte Schnittstellen, letztere haben keine direkten Schnittstellen zum Kunden. Diese Unterscheidung unterstützt den Modellierer bei der Frage nach dem Detaillierungsgrad zu modellierender Kernprozesse in Abhängigkeit vom Modellierungsziel.

- Schnittstellen von Kernprozessen:
  Das Process Landscaping zeichnet sich dadurch aus, dass Schnittstellen zwischen Prozessen von Beginn ihrer Modellierung an berücksichtigt werden. Sie können bereits auf hohem Abstraktionsniveau identifiziert und dann solange verfeinert werden, bis der Typ der Kommunikation und das Format aller auszutauschenden Objekte vollständig festgelegt sind.

- Grundsätze ordnungsmäßiger Modellierung [Ros96]:
  Während die obigen Grundsätze sich konkret auf die Modellierung von Prozessmengen und ihren Schnittstellen beziehen, bilden die Grundsätze der ordnungsmäßigen Modellierung einen allgemeingültigen Ordnungsrahmen zur Darstellung von Inhalten und Gestaltung von Prozessmodellen. Diese Grundsätze werden auch vom Process Landscaping berücksichtigt. Sie unterstützen die Handhabung komplexer, zusammenhängender Prozessmodelle und beinhalten im Einzelnen [BRS95]

  – den Grundsatz der syntaktischen und semantischen Richtigkeit inklusive sichtenübergreifender Konsistenzregeln,
  – den Grundsatz der Relevanz, nach dem die in einem Modell enthaltenen Elemente genau dann relevant sind, wenn ihr Fehlen den Nutzen des Modells bezüglich eines konkreten Modellierungsziels vermindern würde,
  – den Grundsatz der Wirtschaftlichkeit, der dem Detaillierungsgrad eines Modells eine obere Grenze setzt,

- den Grundsatz der Klarheit bezüglich des Layouts von Prozessmodellen und der Findung eines geeigneten Abstraktionsgrades,
- den Grundsatz der syntaktischen und semantischen Vergleichbarkeit, wobei sich die syntaktische Vergleichbarkeit auf die Kompatibilität konsistenter Modelle bezieht und die semantische Vergleichbarkeit die inhaltliche Vergleichbarkeit z.B. von Soll- und Ist-Modellen betrachtet und
- den Grundsatz des systematischen Aufbaus, der dem Sachverhalt der Modellierung in getrennten Sichten und der daraus folgenden Notwendigkeit der Integration der einzelnen Sichten Rechnung trägt.

Das Analyseziel des Process Landscaping ist die Unterstützung der Prozessverantwortlichen durch Aussagen über bestimmte Eigenschaften einer Prozesslandschaft. Dazu werden sowohl statische Analysen mittels Zählen, Aggregieren und Auswerten definierter Attributwerte durchgeführt als auch dynamische Analysen durch Simulationen, die zeitliche Aspekte und Ablaufverhalten eines Modells berücksichtigen. Zur Überprüfung der Analyseergebnisse werden teils Soll- und Ist-Ausprägungen einer Prozesslandschaft verglichen, teils werden Metriken für eine Bewertung von Teilzielen definiert (vgl. Abschnitt 2.3). Die konkret betrachteten Eigenschaften beziehen sich auf Kommunikationsstruktur und -verhalten. Die Analyse dieser Eigenschaften einer Prozesslandschaft mit der Methode des Process Landscaping berücksichtigt die folgenden Kriterien:

- Analysieren heißt messen und bewerten:
  Diesem Prinzip folgt auch die Methode des Process Landscaping. Wie bereits im Begriff *Messen* eine Zwecksetzung notwendig impliziert ist (Messen ist „eine Zuordnung von Zahlen zu Objekten oder Ereignissen“ [Ort83] nach bestimmten Regeln zum Zwecke einer (vergleichenden) Beschreibung oder zur Beurteilung eines Sachverhaltes), so setzt auch das Process Landscaping immer ein Optimierungsziel bzw. eine Änderungsidee voraus, bezüglich derer die Eigenschaften einer Prozesslandschaft zu bewerten sind. Beispielsweise kann die Kommunikation von Prozessen innerhalb einer Prozesslandschaft bezüglich der Änderungsidee betrachtet werden, eine Teilmenge von Aktivitäten an einem gemeinsamen Standort konzentrieren zu wollen. Die für diese Betrachtung angewendete Messmethode kann dabei eine direkte Zuordnung von Zahlen zu Landschaftsbereichen bzw. eine Klassifikation über unterschiedlich komplexe Berechnungsalgorithmen oder eine Durchführung von Simulationsläufen mit anschließenden stochastischen Berechnungen auf der Basis der gewonnenen Simulationsdaten sein. Ein Vergleich der gemessenen Kommunikationseigenschaften vor und nach der Änderung der Prozesslandschaft erlaubt eine anschließende Bewertung dieser Änderungsidee.

- Kommunikationskosten:
  Das Process Landscaping unterscheidet bei der Betrachtung von Kommunikationskosten zwischen den Kosten, die durch Kommunikation innerhalb eines

Standortes (*interne* Kommunikation) entstehen und den Kosten, die durch Kommunikation zwischen verschiedenen Standorten (*externe* Kommunikation) entstehen. Dabei wird – in Anlehnung an viele typische Situationen in der Praxis – angenommen, dass externe Kommunikation teurer zu bewerten ist als interne. Beispiele für externe Kommunikation sind eine Wählleitung mit anschließendem Datenaustausch per WAN oder eine Videokonferenz. Interne Kommunikation kann dagegen beispielsweise durch einen Datenaustausch per Dateizugriff innerhalb eines LANs oder direkt mit einem Gesprächspartner innerhalb eines Raumes stattfinden. Interne Kommunikation ist neben einer schnelleren Abwicklung häufig auch dadurch preiswerter, dass keine Kommunikationssysteme benötigt werden, die zusätzliche Kosten verursachen.

- effiziente Kommunikation:
  Ein globales Optimierungsziel des Process Landscaping ist das Erreichen einer effizienten Kommunikation. Im Rahmen dieser Arbeit wird eine Kommunikation als effizient bezeichnet, wenn ein Informationsaustausch in möglichst geringer Zeit und mit möglichst geringem Aufwand für die beteiligten Kommunikationspartner durchgeführt werden kann. Die Kommunikationseffizienz bezieht sich dabei auf räumliche Verteilung, Auslastung und Informationsfluss und -verteilung von Prozessen [SW02a]. Bei einer ungleichmäßigen Verteilung entstehen sowohl für den auf eine Antwort wartenden als auch für den angesprochenen Kommunikationspartner temporäre und monetäre Aufwände. Je gleichmäßiger eine Verteilung der Kommunikationsaufwände auf die einzelnen Kommunikationspartner in einer Prozesslandschaft gelingt, desto effizienter kann die Kommunikation innerhalb dieser Prozesslandschaft erfolgen. Dabei wird das auch in anderen Bereichen häufig angewandte Verursacherprinzip zugrundegelegt. Dieses legt einem Prozess, der angefragte Informationen versendet, die entstehenden Kommunikationskosten zur Last, was seine Effizienz verringert.

2.2 Methodische Schritte zur Erstellung einer Prozesslandschaft

Ein Ziel des Process Landscaping ist die Modellierung einer Prozesslandschaft, die eine Menge zusammengehöriger Prozesse und ihre Beziehungen zueinander beschreibt. Die notwendigen Schritte zur Erreichung dieses Ziels sind [GW00a, GW00b]:

1. Identifikation von (Kern-)Aktivitäten und ihren Schnittstellen
2. Verfeinerung von Aktivitäten bis auf Prozessmodellebene
3. Verfeinerung von Schnittstellen
4. Verfeinerung von Prozessmodellen
5. Validation der modellierten Prozesslandschaft

Diese Schritte werden im Folgenden zusammen mit den beteiligten Rollen und unterstützenden Techniken vorgestellt. Die Reihenfolge der einzelnen Schritte wird bis auf den ersten nicht fest vorgeschrieben. Diese Freiheit erlaubt unterschiedliche Abstraktionsgrade bei der Modellierung von Aktivitäten und Schnittstellen. Sie unterstützt die Konzentration auf eine den projektspezifischen Anforderungen angepasste Detaillierung dort, wo sie erforderlich ist. Der erste Schritt besteht aus der Identifikation der Kernaktivitäten und ihren Schnittstellen sowie ihrer Anordnung auf der obersten Ebene der zu modellierenden Prozesslandschaft gemäß den in Abschnitt 2.1 erläuterten Prinzipien. Auf jede Kernaktivität kann die Methode des Process Landscaping erneut angewandt werden, bis schließlich individuelle Prozessmodelle und ihre Schnittstellen entstanden sind (vgl. Schritte 2 und 3). Anschließend können Details überall dort hinzugefügt werden, wo es notwendig erscheint (vgl. Schritt 4). Schritt 5 befasst sich nicht mehr mit der Entwicklung, sondern mit der abschließenden Überprüfung einer entstandenen Prozesslandschaft vor ihrer Freigabe.

Die einzelnen Schritte werden in den Abschnitten 2.2.2 bis 2.2.6 entlang eines Beispiels detailliert beschrieben. Für die Diskussion des Process Landscaping wurde bewusst ein komplexes Beispiel der komponentenbasierten Softwareentwicklung gewählt, da sich darin viele typische Abläufe und charakteristische Eigenschaften abbilden lassen, die bei realen Softwareentwicklungsprojekten unter Umständen nur vereinzelt auftreten.

2.2.1 Beispielprozesslandschaft der komponentenbasierten Softwareentwicklung

Zur Erläuterung der einzelnen methodischen Schritte wird eine Softwareprozesslandschaft modelliert, die Aktivitäten, Abläufe und Informationsobjekte der komponentenbasierten Softwareentwicklung abbildet. Die Motivation für die Wahl einer solchen Softwareprozesslandschaft liegt im aktuellen Trend begründet, der komponentenbasierten Softwareentwicklung einen wachsenden Stellenwert beizumessen [FRS+01, SWR+00]. Sie berücksichtigt die Aspekte der flexiblen Verteilung von Anwendungen ebenso wie „die Tendenz, Software aus vordefinierten Bausteinen zusammenzubauen, die immer mehr Anwendungswissen umfassen“ [GT00].

Die Beispielprozesslandschaft ist auf Grundlage von Interviews und Literaturrecherchen entstanden. Für Letzteres sei insbesondere eine Fallstudie eines mehrjährigen realen Softwareprojektes in [AF98] erwähnt.

Die Prozesse der komponentenbasierten Softwareentwicklung beinhalten immer

- das Projektmanagement,
- das Application Engineering,
- das Component Engineering,
- das Qualitätsmanagement und
- das Domain Engineering.

Für die Diskussion der Modellierung und Analyse dieser Kernprozesse sind die englischen Begriffe gewählt worden, wenn keine prägnante Übersetzung möglich ist, wie beispielsweise für den Begriff des Domain Engineering, der nicht ohne ausführliche Erläuterung des übersetzten Begriffs auskommt. Des Weiteren hilft die Verwendung des englischen Begriffs Application Engineering, zwischen der Anwendungsentwicklung im Allgemeinen und der Anwendungsentwicklung im Rahmen der komponentenbasierten Softwareentwicklung – üblicherweise als Application Engineering bezeichnet – zu unterscheiden.

Alle Schritte des Process Landscaping werden nachfolgend entlang des Beispiels der komponentenbasierten Softwareentwicklung erläutert, indem die zugehörigen Kernprozesse zunächst als Kernaktivitäten in einem Koordinatensystem angeordnet und anschließend sowohl Aktivitäten als auch ihre Schnittstellen nach und nach verfeinert werden. Die resultierende Softwareprozesslandschaft dient als Basis für die Erläuterung und Validation der Analyseschritte und für die Diskussion der formalen Notation einer Prozesslandschaft.

2.2.2 Identifikation von Kernaktivitäten und ihren Schnittstellen

Der erste Modellierungsschritt des Process Landscaping hat die Erstellung der obersten Ebene einer Prozesslandschaft zum Ziel. Diese Ebene beinhaltet die Kernaktivitäten und ihre Schnittstellen zueinander. Die für den Durchlauf eines Kundenlebenszyklus notwendigen Kernaktivitäten und ihre Schnittstellen werden im Rahmen von Interviews mit den Prozessverantwortlichen identifiziert und in einem Koordinatensystem angeordnet (vgl. Abbildung 2.1). Als unterstützende Techniken können verschiedene Methoden der Wissensakquise eingesetzt werden. Hier eignen sich insbesondere freie und standardisierte Interviews [Kor00]: Während ein freies Interview individuell und lediglich anhand eines (groben) Leitfadens durchgeführt wird, werden in einem standardisierten Interview vorformulierte Fragen gestellt, um sich eine bereits existierende Vorstellung eines (geistigen) Modells erläutern oder bestätigen zu lassen.

Die Gruppierung der identifizierten Kernaktivitäten auf der obersten Abstraktionsebene wird durch ein Koordinatensystem unterstützt. Auf der X-Achse werden die Kernaktivitäten in Abhängigkeit ihrer direkten Schnittstellen zum Kunden angeordnet: Je mehr direkte Schnittstellen ein Kernprozess aufweist, desto weiter links wird die zugehörige Kernaktivität angeordnet. Interne, rein unterstützende Kernaktivitäten werden rechts angeordnet. Die Y-Achse beschreibt, an welcher Stelle des Kundenlebenszyklus die beschriebenen Prozesse benötigt werden: Je früher ein Prozess im Verlauf eines Kundenlebenszyklus angestoßen wird, desto weiter unten wird er angeordnet. Unterstützung bei der Anordnung der Kernaktivitäten im Koordinatensystem bietet auch hier das freie Interview, wenn vom Interviewpartner eine reale Problemlösung nachvollzogen wird. Die durchzuführenden Kernprozesse werden in der Reihenfolge ihres Auftretens aufgelistet und vom Modellierer entsprechend im Koordinatensystem der Prozesslandschaft (als Kernaktivitäten) angeordnet.

Gemäß den beiden Achsen des Koordinatensystems ergibt sich eine natürliche Rich-

tung für die Modellierung der Prozesslandschaft. Sie beginnt mit dem chronologisch ersten Kundenprozess (links unten). Von diesem ausgehend werden Prozesse modelliert, die ihn direkt unterstützen sowie Prozesse, die zeitlich unmittelbar folgen. Die Modellierungsrichtung gewährleistet, dass Details über andere Prozesse (z.B. von der rechten oberen Ecke der Prozesslandschaft) nicht beschrieben werden, bevor sie nicht aktuell benötigt werden. Durch die Modellierungsrichtung ergibt sich eine schrittweise Betrachtung aller in der Prozesslandschaft dargestellten Prozesse.

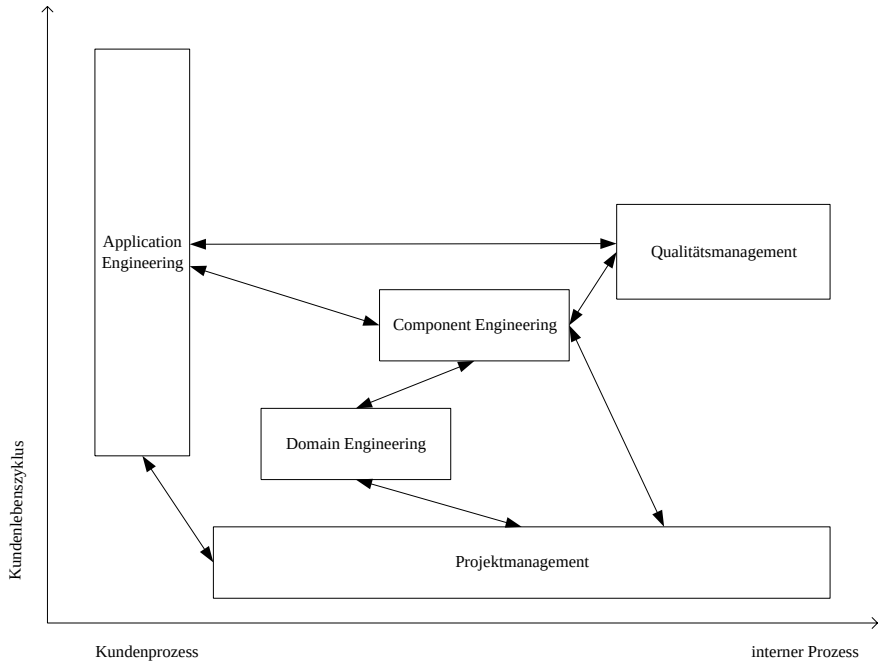


Abbildung 2.1: Oberste Ebene einer Prozesslandschaft zur Darstellung komponentenbasierter Softwareentwicklung

Abbildung 2.1 zeigt die Anordnung der identifizierten Kernaktivitäten auf der obersten Ebene der Prozesslandschaft als Ergebnis des ersten Schrittes des Process Landscaping. Die Ebene besteht aus fünf Kernaktivitäten und ihren Schnittstellen. Die Anordnung der einzelnen Kernaktivitäten lässt erkennen, dass hier ein erster Kundenkontakt über Prozesse des Projektmanagement stattfindet. Diese Kernaktivität stellt innerhalb der in Abbildung 2.1 dargestellten Prozesslandschaft eine Besonderheit dar. Prozesse aus dem Projektmanagement betreffen sowohl kundennahe als auch rein interne, unterstützende Aktivitäten. Aus diesem Grund ist diese Kernaktivität entlang der gesamten X-Achse angeordnet, was jedoch nicht bedeuten soll, dass ihre Aktivitäten nur zu Beginn eines Kundenlebenszyklus vorkommen können (vgl. z.B. die Schnittstelle zwischen Application Engineering und Projektmanagement). Aus Übersichtlichkeitsgründen erstreckt sich das Projektmanagement jedoch nicht über die gesamte Fläche

des Koordinatensystems, sondern bleibt nah an der X-Achse.

Um die Entwicklung einer vom Kunden gewünschten Software durchführen zu können, müssen u.a. vom Qualitätsmanagement vorgegebene Richtlinien berücksichtigt werden. Die zugehörigen Prozesse haben im Vergleich zum Projektmanagement keine direkten Schnittstellen zum Kunden. Daher ist beispielsweise die Kernaktivität *Qualitätsmanagement* – gemäß dem Prinzip der Unterteilung von Kernprozessen in Kundenprozesse und interne Prozesse – auf der obersten Ebene der Prozesslandschaft weiter rechts angeordnet. Der Prozess des Application Engineering setzt die Software aus den vorhandenen Komponenten zusammen. Er hat im Gegensatz zum Qualitätsmanagement viele Berührungspunkte zum Kunden. Da er sich fast über den gesamten Kundenlebenszyklus erstreckt, ist er entlang der Y-Achse angeordnet. Das Component Engineering ist eng mit dem Application Engineering verzahnt, da hier Komponenten entwickelt werden, die dem Application Engineering nicht oder nicht in ausreichender Qualität zur Verfügung stehen. Das Component Engineering hat aber keine direkten Schnittstellen zum Kunden und ist daher im Koordinatensystem etwas weiter rechts angeordnet.

Die Schnittstellen zwischen den Kernaktivitäten sind auf dieser obersten Ebene durch bidirektionale Pfeile angedeutet. Sie können identifiziert werden, indem die Prozessverantwortlichen nach den Typen der auszutauschenden Informationen befragt werden. Wenn die Prozessverantwortlichen zweier miteinander in Beziehung stehender Kernprozesse befragt werden, lassen sich unterschiedliche Auffassungen und Gewichtungen bezüglich ihrer Zusammenarbeit erkennen und in einer gemeinsamen Abstimmung klären.

Der Umfang der im Koordinatensystem platzierten Kernaktivitäten steht nicht unbedingt in einem direkten Bezug zu ihrer Komplexität. Daher lässt sich von ihrer Größe weder auf die Menge der sie enthaltenden Teilaktivitäten noch auf ihren Stellenwert innerhalb der gesamten Prozesslandschaft schließen. Die in Abbildung 2.1 verwendeten Größenverhältnisse sind lediglich aus Gründen der Übersichtlichkeit gewählt und vermeiden Überschneidungen der Rechtecke und Pfeile. Die abgebildete oberste Ebene der Prozesslandschaft unterstützt jedoch die zusammenhängende Betrachtung der beteiligten (später verfeinerten) Kernaktivitäten und erfüllt damit die zu Beginn des Abschnitts 2.1 entsprechend formulierte Anforderung.

2.2.3 Verfeinerung von Kernaktivitäten bis auf Prozessmodellebene

Kernaktivitäten stellen die Kernprozesse zunächst nur sehr abstrakt dar. Neben einem Namen wird lediglich angegeben, zu welchen anderen Kernaktivitäten Schnittstellen bestehen. Kernaktivitäten werden vom Prozessmodellierer verfeinert, um zusätzliche Details über die Kernprozesse in die Prozesslandschaft aufzunehmen. Ziel ist es, eine vollständige Liste aller Aktivitäten innerhalb eines Kernprozesses zu erarbeiten, deren Beschreibung bei Bedarf weiter verfeinert werden kann. Hierfür empfehlen sich weitere Methoden der Wissensakquise wie z.B. die offene, direkte Beobachtungsmethode [Kra96], die noch weiter in die strukturierte und unstrukturierte Beobachtung unter-

teilt wird: Während bei der strukturierten Beobachtungsmethode Arbeitsabläufe der zu modellierenden Prozesse betrachtet und nach im voraus festgelegten Richtlinien protokolliert werden, liegen bei der unstrukturierten Beobachtung keine Richtlinien zur Erhebung der beobachteten Abläufe vor.

Die Verfeinerung einer Kernaktivität geschieht durch die Angabe der zugehörigen Aktivitäten inklusive ihrer Abhängigkeiten, also durch die Erstellung einer neuen Prozesslandschaft pro Kernprozess. Die Aktivitäten können ihrerseits so lange verfeinert werden, bis sie aus Sicht des Modellierers nicht weiter zerlegt werden sollten. Ablaufinformationen, wie die Festlegung einer Durchführungsreihenfolge der identifizierten Aktivitäten, werden im Rahmen dieses Methodenschrittes nicht modelliert, da sie die Übersichtlichkeit bezüglich der wichtigsten Aktivitäten innerhalb der Kernaktivitäten einschränken. Solche Informationen werden erst bei einer weiteren Verfeinerung der Aktivitäten berücksichtigt (siehe Abschnitt 2.2.5).

Um auch nach durchgeführten Verfeinerungen das Gesamtbild einer Prozesslandschaft nicht aus den Augen zu verlieren ist es wichtig, eine Übersicht über die Hierarchiebeziehungen zwischen Aktivitäten einer Landschaft zu erstellen, die sich auf unterschiedlichen Abstraktionsebenen befinden. Diese Übersicht, die einfach als Baumstruktur abgebildet werden kann, gewährleistet zusammen mit der Darstellung der Kernaktivitäten auf der obersten Ebene einen Überblick sowohl in die Breite als auch in die Tiefe einer modellierten Prozesslandschaft.

Aufgrund der Unterscheidung zwischen Aktivitäten mit und ohne Ablaufinformationen wird ihm Rahmen dieses Methodenschrittes bewusst von *Verfeinerung bis auf Prozessmodellebene* gesprochen. Der Begriff *Aktivität* wird hier für modellierte Prozesse verwendet, bei denen vom Ablauf abstrahiert sein kann, während der Begriff *Prozessmodell* die Berücksichtigung von Ablaufinformationen impliziert (vgl. Abschnitt 1.3). Bezogen auf die gesamte Prozesslandschaft wird entsprechend von ihren *oberen Ebenen* gesprochen, die keine Ablaufinformationen enthalten, während in ihren *unteren Ebenen* diese explizit berücksichtigt sind.

Die Abstraktion von Ablaufinformationen hat neben einer besseren Übersichtlichkeit vor allem den Vorteil, dass für die oberen Ebenen einer Prozesslandschaft ein Gesamtbild erstellt werden kann, welches auch der Durchführung unstrukturierter und flexibler Prozesse als Unterstützung dient, indem die wichtigsten Aktivitäten und existierende Relationen zueinander dargestellt sind. Flexibilität bezieht sich hier auf die Ablaufreihenfolge definierter Aktivitäten, die sich aus der Unstrukturiertheit des zugrundeliegenden Prozesses ergibt. Eine korrespondierende Flexibilität in der Prozesslandschaft kann durch das Abstrahieren von einem festen Ablauf ermöglicht werden. Durch die Möglichkeit des Abstrahierens von Ablaufinformationen wird dem Grundsatz der Wirtschaftlichkeit genüge getan, der dem Detaillierungsgrad eines Modells insbesondere für unstrukturierte Prozesse eine obere Grenze setzt (vgl. Abschnitt 2.1).

Abbildung 2.2 beinhaltet ein Gesamtbild ohne Ablaufinformationen für die Prozesslandschaft der komponentenbasierten Softwareentwicklung. Mit fünf Aktivitäten auf der obersten Abstraktionsebene ist die Breite der Prozesslandschaft festgelegt, während die Tiefe bislang durch drei Hierarchiestufen begrenzt ist. Letztere kann durch

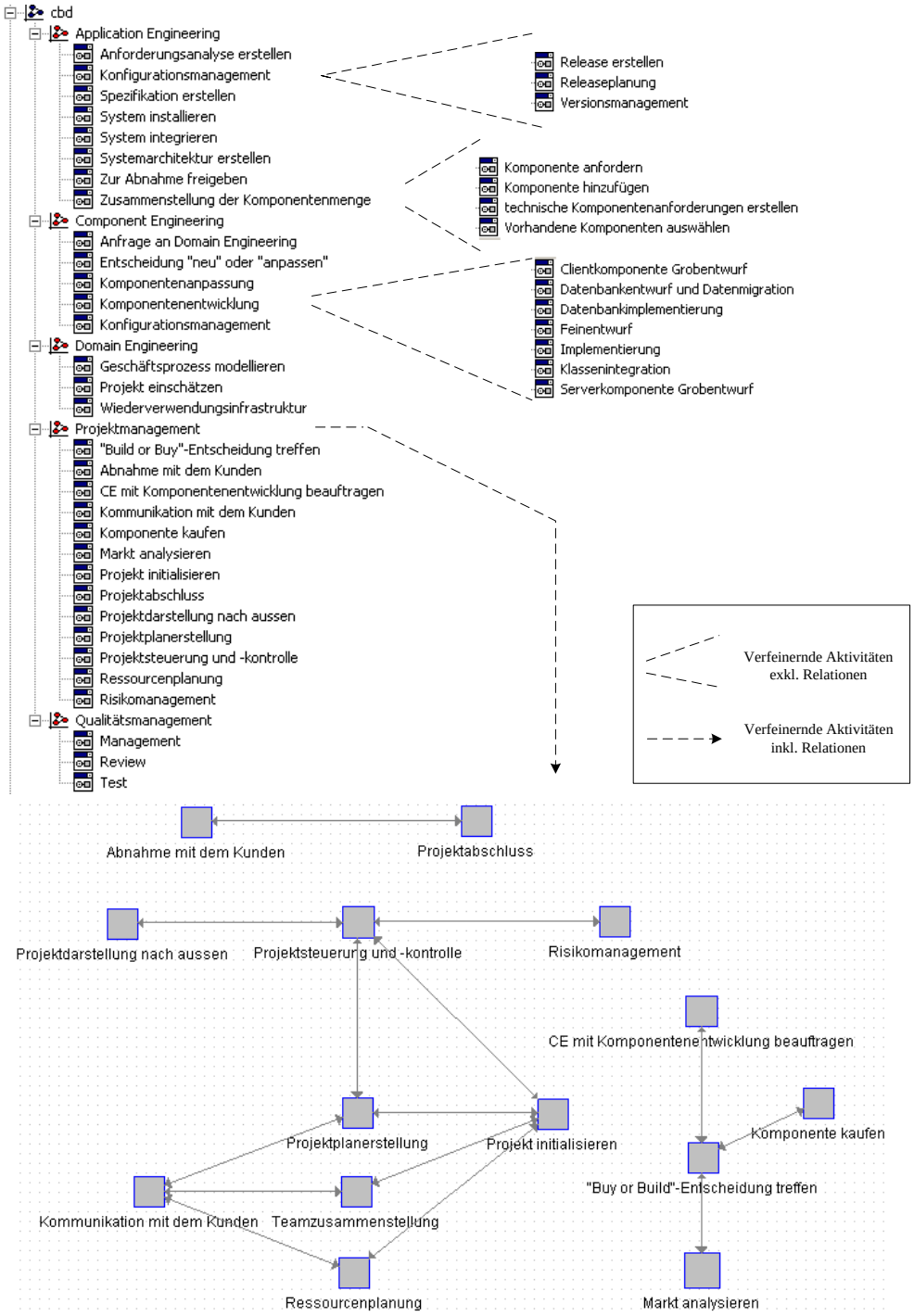


Abbildung 2.2: Verfeinerung von Kernaktivitäten bis auf Prozessmodellebene

weitere Verfeinerungen der dargestellten Aktivitäten verändert werden.

Der obere Teil der Abbildung zeigt die Kernaktivitäten und Aktivitäten der Softwareprozesslandschaft als Baumstruktur. Die Kernaktivitäten, die in Abbildung 2.1 als oberste Ebene modelliert sind, erkennt man hier an den Symbolen, die das Koordinatensystem einer Prozesslandschaft darstellen. Sie sind den betreffenden Aktivitätsnamen vorangestellt. Die eckigen Symbole repräsentieren verfeinernde Aktivitäten, die selbst auch wieder verfeinert sein können. Die Aktivität *Komponentenentwicklung* innerhalb der Kernaktivität *Component Engineering* ist zum Beispiel durch eine Menge von sieben Aktivitäten weiter verfeinert worden. Die Aktivitäten *Konfigurationsmanagement* und *Zusammenstellung der Komponentenmenge*, die beide Teilbereiche des Application Engineering abdecken, sind ebenfalls bereits weiter verfeinert worden. In der Abbildung sind diese Verfeinerungen aus Platzgründen durch gestrichelte Linien angedeutet, die die Aktivitäten mit ihren jeweiligen Teilaktivitäten verbinden.

Am Beispiel der verfeinerten Aktivität *Projektmanagement* wird im unteren Teil der Abbildung 2.2 gezeigt, ob Relationen zu anderen Aktivitäten auf einer gemeinsamen Abstraktionsebene existieren und wie deren Existenzen dargestellt werden. Der gestrichelte Pfeil verbindet die Aktivität *Projektmanagement* mit ihrer verfeinerten Darstellung. Die bidirektionalen Pfeile zwischen den einzelnen Aktivitäten innerhalb der Verfeinerung weisen darauf hin, dass die dadurch angedeuteten Schnittstellen noch nicht verfeinert sind (vgl. Abschnitt 2.2.4). Die abgebildeten Aktivitäten sind auch in der Baumstruktur dargestellt, jedoch ohne Informationen über vorhandene Schnittstellen untereinander.

2.2.4 Verfeinerung von Schnittstellen

Auf dem obersten Abstraktionsniveau einer Schnittstelle wird zunächst nur angezeigt, dass die beteiligten Prozesse Aktivitäten enthalten, die mindestens eine Schnittstelle haben. Für eine detailliertere Beschreibung werden diese Schnittstellen vom Prozessmodellierer mit jeweils einem Objekttyp assoziiert (im Beispiel zunächst über entsprechende Namen, vgl. Abbildung 2.3), der angibt, welche Art von Information ausgetauscht werden kann. Die Identifikation der in einem Softwareentwicklungsprojekt benötigten und erzeugten Dokumente ist hier hilfreich bei der Wissensakquise. Dabei geht man davon aus, dass die wichtigsten Informationen in Form von Dokumenten ausgetauscht werden (z.B. Anforderungsdokument, Projektplan, Fehlerreport, usw.).

Bei der Verfeinerung von Schnittstellen werden jedoch nicht nur die auszutauschenden Objekttypen identifiziert. Für jeden Objekttyp wird gleichzeitig auch die Richtung des Informationsaustausches definiert. Damit besteht das Ergebnis der Verfeinerung von Schnittstellen aus einer Menge von Objekttypen inklusive ihrer Relationen zu Aktivitäten.

Abbildung 2.3 zeigt das Beispiel der verfeinerten Schnittstelle zwischen den Kernprozessen *Qualitätsmanagement* und *Application Engineering*. Sie beinhaltet insgesamt fünf Schnittstellen zusammen mit den assoziierten Objekttypen, die ausgetauscht wer-

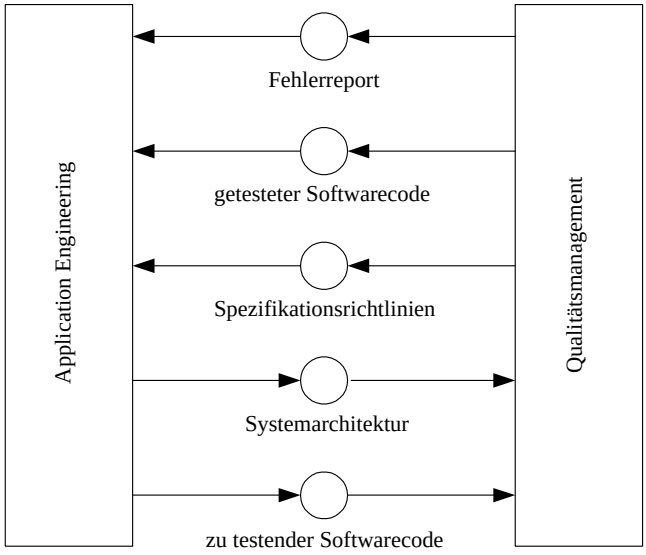


Abbildung 2.3: Verfeinerte Schnittstelle zwischen Qualitätsmanagement und Application Engineering

den. In der grafischen Repräsentation werden die Schnittstellen als Kreise dargestellt. Gerichtete Kanten zeigen die Richtung des Informationsaustausches an. Beispielsweise werden dem Application Engineering Spezifikationsrichtlinien seitens des Qualitätsmanagements zur Verfügung gestellt. Einige Objekttypen werden wechselseitig zwischen Prozessen des Application Engineering und Prozessen des Qualitätsmanagements ausgetauscht. Dies gilt z.B. für den Softwarecode, der als zu testender dem Qualitätsmanagement zur Verfügung gestellt wird und nach einer Testphase zusammen mit einem Fehlerreport an das Application Engineering zurückgeht.

2.2.5 Verfeinerung von Prozessmodellen

Prozessmodelle stellen typischerweise den konkreten Ablauf einer Menge von Aktivitäten dar und beinhalten damit explizit Ablaufinformationen (vgl. Abschnitt 1.3). Sie sind daher auf den unteren Ebenen einer Prozesslandschaft anzuordnen. Die Wissensakquise über die einzelnen Aktivitäten und die Reihenfolge ihrer Durchführung startet mit weiteren Interviews und dem sorgfältigen Lesen verfügbarer Dokumentation. Indirekte Methoden der Wissensakquise eignen sich als Ergänzung der in den Abschnitten 2.2.2 und 2.2.3 vorgeschlagenen direkten Methoden, die bei der Identifikation von Kernaktivitäten und ihren Schnittstellen und ihrer Verfeinerung bis auf Prozessmodellebene zum Einsatz kommen (vgl. Abschnitt 2.2.2). Sie erfragen Wissen nicht direkt, sondern versuchen, die gesuchte Information aus anderen Informationen abzuleiten. Dazu bedienen sie sich verschiedener Skalierungstechniken wie beispielsweise dem Konstruktgitter-Verfahren (repertory grid) [SG87].

Eine weitere wichtige Methode zur Gewinnung detaillierterer Informationen über Aktivitäten ist die Entwicklung von Szenarien, die typische Situationen beschreiben. Dabei werden gleichzeitig weitere Aktivitäten identifiziert, die ebenfalls mit dieser Situation in Verbindung stehen. Jedes Szenario startet mit einer abstrakten Beschreibung. Jeweils ein Prozessverantwortlicher und ein Prozessmodellierer nutzen das Szenario für einen schrittweisen Walkthrough, wobei jeder Aktivität Detailwissen hinzugefügt wird (etwa mögliche Ausnahmesituationen, die auftreten können; welches Objekt als Eingabe benötigt wird und welches als Ergebnis produziert wird; wer verantwortlich ist; welche Werkzeuge eingesetzt werden; welche Richtlinien beachtet werden müssen). Das Ergebnis eines Walkthrough ist ein detaillierteres Prozessmodell, das auf die gleiche Art und Weise wie die gerade beschriebene noch weiter verfeinert werden kann.

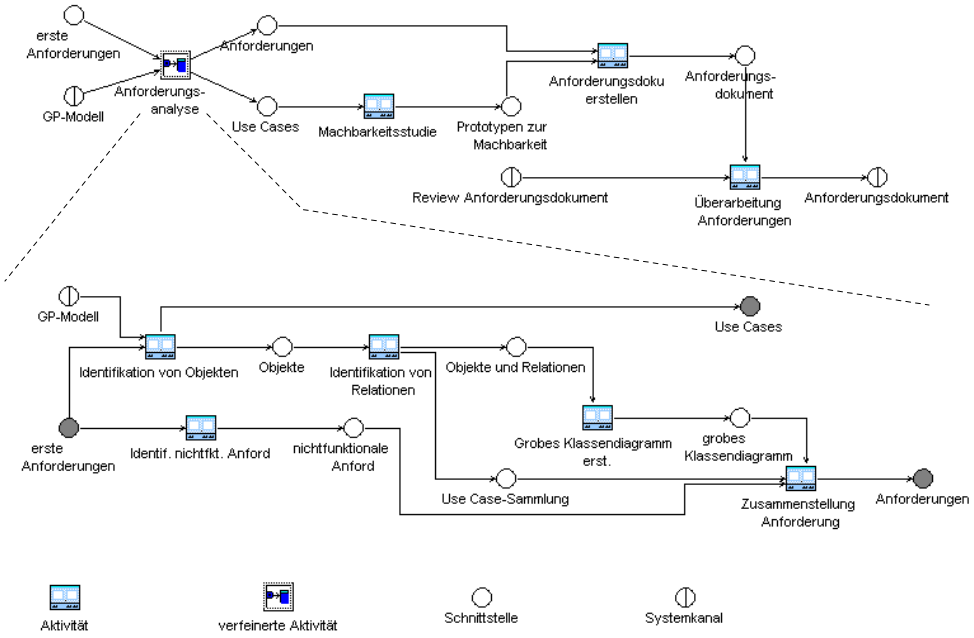


Abbildung 2.4: Prozessmodell zur Beschreibung der Anforderungsanalyse

Abbildung 2.4 zeigt ein Prozessmodell zur Beschreibung der Aktivität *Anforderungsanalyse erstellen*, hier dargestellt als Funsoft-Netz [Gru91], einer Variante der High-Level-Petrinetze. Wie die Legende zeigt, sind die einzelnen Teilaktivitäten in der Abbildung durch rechteckige Symbole dargestellt. Die verschiedenen Schnittstellen, die die zu verarbeitenden Informationsobjekte weiterleiten, sind durch Kreise gekennzeichnet. Auf einer tieferen Abstraktionsebene weiter verfeinerte Aktivitäten sind am rechteckigen Symbol mit einem breitem Rand erkennbar. Werden Informationsobjekte über Prozessmodell-Grenzen hinweg ausgetauscht, ist dies durch Kreise mit senkrecht durchgezogener Linie (sogenannte Systemkanäle) symbolisiert. In Abbildung 2.4 wird

beispielsweise der Review-Report zum Anforderungsdokument vom Qualitätsmanagement über den Systemkanal *Review Anforderungsdokument* zur Aktivität *Überarbeitung Anforderungen*, einer Aktivität innerhalb der Anforderungsanalyse, weitergeleitet.

Das abgebildete Prozessmodell ist der Kernaktivität *Application Engineering* zugeordnet und stellt das – immer noch recht grobe – Szenario einer objektorientierten Anforderungsanalyse dar. Ausgehend von den ersten Anforderungen an das zu entwickelnde Softwaresystem und einem Geschäftsprozessmodell des Kundenunternehmens erfolgt die Anforderungsanalyse zusammen mit einer Machbarkeitsstudie. Letztere wird auf der Basis von Use Cases erstellt. Sie gibt Auskunft darüber, ob bestimmte technische Anforderungen erfüllt werden können und ob sich vielleicht einige der Anforderungen widersprechen. Die Erkenntnisse aus der Machbarkeitsstudie und die identifizierten Anforderungen werden in einem Anforderungsdokument zusammengefasst und nach einem Review durch das Qualitätsmanagement entweder zunächst überarbeitet oder nach erfolgreichem Review an nachfolgende Aktivitäten, wie zum Beispiel *Spezifikation erstellen*, weitergeleitet.

Die Aktivität *Anforderungsanalyse* ist noch einmal explizit verfeinert worden. Diese Verfeinerung ist im unteren Teil der Abbildung 2.4 dargestellt und durch gestrichelte Linien mit der verfeinerten Aktivität verbunden. Die Aktivität *Anforderungsanalyse* besteht in diesem Beispiel aus Teilschritten zur Identifikation nichtfunktionaler Anforderungen und zu berücksichtigender Objekttypen. Für Letztere werden in einem nächsten Schritt bestehende Relationen identifiziert, Use Cases und ein erstes grobes Klassendiagramm (zunächst noch ohne vollständige Benennung und Zuordnung von Methoden) erstellt. Alle Teilergebnisse werden zu einer Menge von Anforderungen zusammengestellt, die – wie im oberen Teil der Abbildung 2.4 erkennbar – mit den Erkenntnissen der Machbarkeitsstudie zum gesamten Anforderungsdokument zusammengeführt werden. Für die zugrundeliegende formale Definition der Verfeinerung einer durch Petrinetze dargestellten Prozesslandschaft sei an dieser Stelle auf Kapitel 3 verwiesen.

Werden Informationsobjekte zwischen zwei Aktivitäten ausgetauscht, die zwei unterschiedlichen Kernaktivitäten A und B zugeordnet sind, so gehören die entsprechenden Systemkanäle zur Schnittstellenvereinbarung zwischen A und B. Der Name eines Systemkanals gibt den Typ der auszutauschenden Informationsobjekte an. Eine vollständige Schnittstellenvereinbarung zwischen zwei (bereits verfeinerten) Kernaktivitäten beinhaltet folgende Informationen (vgl. Abbildung 2.3):

- Typen der auszutauschenden Informationsobjekte,
- vollständige Menge der Aktivitäten, die die Schnittstelle zum Versand von Daten nutzen (inklusive der verantwortlichen Rollen),
- vollständige Menge der Aktivitäten, die die Schnittstelle zum Empfang von Daten nutzen (inklusive der verantwortlichen Rollen),
- Medium, über das Daten auszutauschen sind (Applikation, e-mail, Telefon, etc.).

Die Wissensakquise über die einzelnen Prozesse liefert als Ergebnis sowohl detailliertere, verfeinerte Aktivitäten als auch die Beschreibung verfeinerter Schnittstellen zwischen den Kernaktivitäten einer Prozesslandschaft. Im Beispiel in Abbildung 2.4 sind nicht alle Ergebnisse der Wissensakquise detailliert modelliert. Beispielsweise kann die Anforderungsanalyse durch eine ausführlichere Darstellung der Zusammenarbeit mit dem Qualitätsmanagement (Reviews der Use Cases und des Klassendiagramms) noch weiter verfeinert werden.

2.2.6 Validation der modellierten Prozesslandschaft

Die Validation einer Prozesslandschaft hat deren Freigabe und Implementation zum Ziel. Für die Freigabe einer Prozesslandschaft müssen alle Details von den Prozessverantwortlichen überprüft werden. Sind sie der Ansicht, dass alle Aktivitäten inklusive ihrer Schnittstellen vollständig und korrekt sind, so ist damit eine Vorbedingung für die Freigabe erfüllt. Die Begriffe der Vollständigkeit und Korrektheit werden hier jedoch nicht im streng mathematischen Sinne verwendet. Eine Prozesslandschaft wird als vollständig angesehen, wenn alle Prozessverantwortlichen die Menge der Aktivitäten und Schnittstellen als ausreichend und ausreichend detailliert erachten, um die modellierten Prozesse mit Hilfe der Prozesslandschaft durchführen zu können. Eine vollständige Prozesslandschaft gilt als korrekt, wenn die Prozessverantwortlichen alle modellierten Relationen zwischen Aktivitäten und Schnittstellen als sinnvoll und erforderlich akzeptieren und sich die Prozessbeteiligten mit den dargestellten Aktivitäten und (falls modelliert) Abläufen identifizieren können.

Unterstützung bei der Überprüfung auf Vollständigkeit bietet die Suche nach Aktivitäten und Informationsobjekten innerhalb einer Prozesslandschaft, die isoliert modelliert sind, also keine Relationen zu anderen Landschaftselementen aufweisen. Bezüglich der Korrektheit der modellierten Relationen zu Informationsobjekten ist zu untersuchen, ob diese nur produziert, aber nicht verwendet werden bzw. umgekehrt von einer Aktivität verwendet, aber von keiner anderen dargestellten Aktivität erzeugt werden. Eine solche Situation sollte nur in Ausnahmefällen vorkommen. In der Regel kann davon ausgegangen werden, dass produzierte Informationsobjekte einen Verwendungszweck innerhalb einer Prozesslandschaft haben bzw. benötigte Informationsobjekte von einer Aktivität erzeugt worden sind.

Eine weitere Vorbedingung für die Freigabe einer Prozesslandschaft besteht in der Vereinbarung der Schnittstellen zwischen Aktivitäten. Dazu wird für jede Aktivität, die Ablaufinformationen enthält, ein Walkthrough mit dem Prozessverantwortlichen des freizugebenden Modells und den Verantwortlichen aller mit den zugehörigen Schnittstellen verbundenen Modellen durchgeführt. Dieser wird vom Prozessverantwortlichen der betreffenden Aktivität moderiert. Ziele eines solchen Walkthrough sind die Präsentation von Prozessdetails zur Erlangung eines gemeinsamen Verständnisses für die Struktur und den Ablauf des Prozesses sowie eine abschließende Vereinbarung der Struktur aller das Modell betreffenden Schnittstellen. Nach Abschluss dieser Vereinbarung für alle Aktivitäten kann die Prozesslandschaft instanziiert werden, indem

alle Aktivitäten bzw. Prozessmodelle inklusive ihrer Schnittstellen instanziiert werden. Die einzelnen Modelle unterliegen ab diesem Zeitpunkt einer Änderungskontrolle. Die Führung einer entsprechenden Änderungshistorie ist sinnvoll, um die Nachvollziehbarkeit geänderter Modelle zu gewährleisten.

Es gibt verschiedene Möglichkeiten, eine Prozesslandschaft in die Praxis umzusetzen. Beispiele hierfür sind das traditionelle Training der Prozessbeteiligten, operationale Tests mit verschiedenen Testszenarien und die Ausarbeitung detaillierter Arbeitsanweisungen. Walkthroughs entlang von Testszenarien überprüfen, ob die Prozesslandschaft die realen Prozesse tatsächlich widerspiegelt. Dabei sollten Szenarien verwendet werden, die mehr als eine verfeinerte Aktivität und mehr als eine Kernaktivität abdecken.

Das Erstellen von Arbeitsanweisungen für die einzelnen Arbeitsschritte aller Prozesse unterstützt die Implementation der Prozesslandschaft auf einem detaillierten Niveau insbesondere für diejenigen Prozesse, die mehrere unterschiedliche Verantwortungsbereiche betreffen, da jede Weiterleitung an einen anderen Prozessbeteiligten explizit aufgeführt ist. Arbeitsanweisungen informieren über Verantwortlichkeiten, Vereinbarungen bezüglich Zeit- und Kostenrahmen (Service Levels), Schlüsselindikatoren über die Performanz von Tätigkeiten und Richtlinien, die für jeden Arbeitsschritt beachtet werden müssen.

Abbildung 2.5 gibt abschließend einen Überblick über das Ergebnis der Anwendung des Process Landscaping auf die Prozesse einer komponentenbasierten Softwareentwicklung. Die entstandene Prozesslandschaft ist relativ abstrakt und nicht vollständig dargestellt, da dies den Rahmen der Abbildung sprengen würde und außerdem der Überblick verloren ginge. Es soll lediglich ein Gefühl für den Aufbau und die Komplexität der verschiedenen Abstraktionsebenen und ihren Relationen zueinander vermittelt werden.

Die Kernaktivitäten der Prozesslandschaft sind als Rechtecke auf der obersten Abstraktionsebene zusammen mit angedeuteten Schnittstellen (bidirektionale Pfeile) zwischen ihnen angeordnet. Des Weiteren sind die Kernaktivitäten *Application Engineering* und *Projektmanagement* (in Abbildung 2.5 mit B und E abgekürzt; vgl. Legende in Abbildung 2.5) auf einer tieferen Abstraktionsebene beispielhaft verfeinert. Diese Verfeinerungen sind durch gestrichelte Linien mit den jeweils verfeinerten Kernaktivitäten, zur besseren Übersichtlichkeit weiß hinterlegt, verbunden. Sie beinhalten nicht nur die Menge der verfeinernden Aktivitäten, sondern zusätzlich auch durch Kreise angedeutete Informationsobjekte, die diese austauschen. Ablaufinformationen sind dabei noch nicht berücksichtigt worden. Dies ist daran erkennbar, dass die zugehörigen Pfeile meist keine eindeutige Richtung eines Informationsaustausches angeben. Abbildung 2.5 zeigt ebenfalls die Verfeinerung einiger Schnittstellen durch

- die explizite Benennung der auszutauschenden Informationsobjekte,
- die Angabe der jeweiligen Richtung des Informationsaustausches und
- die (abstrakte) Darstellung aller beteiligten Aktivitäten, die über die jeweilige Schnittstelle entsprechende Informationen empfangen bzw. versenden.

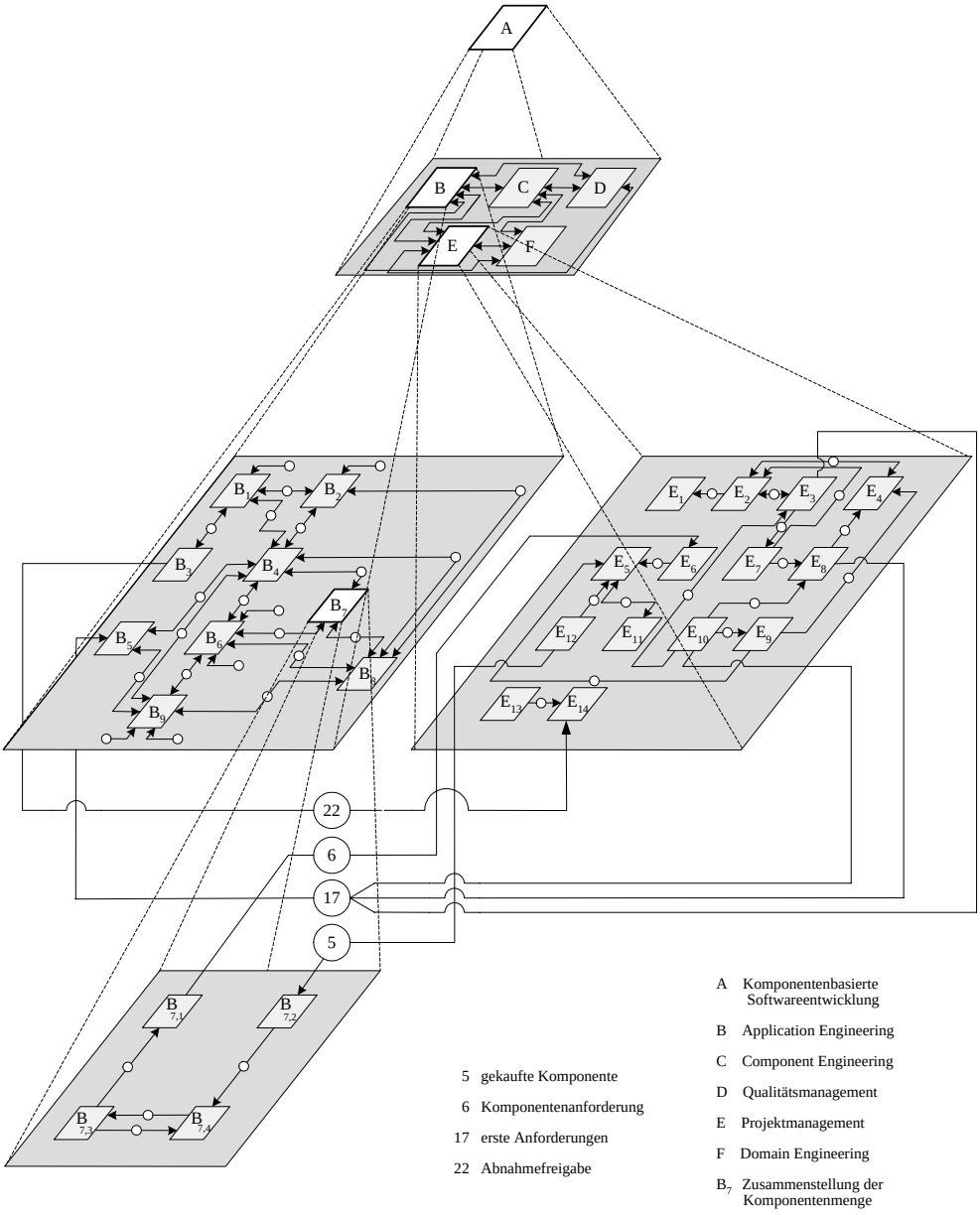


Abbildung 2.5: Gesamtbild der Prozesslandschaft einer komponentenbasierten Softwareentwicklung

Beispielsweise wird die Information über die *Abnahmefreigabe* (22) für ein erstelltes Softwaresystem von einer verfeinernden Aktivität des Application Engineering (B_3) an eine verfeinernde Aktivität des Projektmanagement (E_{14}) gesendet.

Innerhalb des verfeinerten Application Engineering ist beispielhaft eine verfeinernde Aktivität (B_7) nochmals verfeinert. So wird verdeutlicht, dass die Anzahl der Abstraktionsebenen innerhalb einer Prozesslandschaft beliebig und unterschiedlich sein kann. Zusätzlich ist in dieser Abstraktionsebene anhand der überall eindeutigen Richtung des Informationsaustausches erkennbar, dass dort Ablaufinformationen berücksichtigt sind. Ab welchem Detaillierungsgrad dies geschieht, ist jedoch prozessspezifisch und bleibt dem Modellierer überlassen.

Die verschiedenen Arten von Verfeinerungen unterstützen die inkrementelle Vervollständigung von Prozesslandschaften. Da Aktivitäten und Schnittstellen in beliebiger Reihenfolge verfeinert werden können, sind detailliertere Informationen über die verschiedenen Landschaftselemente jederzeit in die Prozesslandschaft integrierbar, ohne dass dabei der Überblick über den Gesamtzusammenhang verloren geht. Die oberen Ebenen einer Prozesslandschaft unterstützen vor allem unstrukturierte Prozesse ohne festgelegten Ablauf, also insbesondere auch Softwareprozesse mit einem hohen Anteil an Kreativität und Flexibilität, die bislang kaum durch Prozessmodellierung unterstützt werden. Der Aufwand hierfür war im Verhältnis zum Nutzen meist zu hoch, wenn Ablaufinformationen von Beginn an im Modell zu berücksichtigen waren.

2.3 Methodische Schritte zur Analyse einer Prozesslandschaft

Weitere Ergebnisse des Process Landscaping, die durch verschiedene Analyseschritte erzielt werden, sind Aussagen über Kommunikationseigenschaften einer verteilten Prozesslandschaft. Nutzer der verschiedenen Analyseergebnisse sind vor allem das Projekt- und Prozessmanagement sowie die Prozessverantwortlichen. Ihre Aufgabe ist es unter anderem, Möglichkeiten zur Effizienzsteigerung der Projektkommunikation zu identifizieren [Ros96]. Dabei ist allerdings davon auszugehen, dass insbesondere für fachlich sinnvoll strukturierte Prozesse die Analyse von Kommunikationseigenschaften lohnt. Dies sind Prozesse, die zumindest als funktionsfähig und zielführend angesehen werden. Bei Prozessen, die diese Kriterien nicht erfüllen, lässt sich zwar auch eine Kommunikationsanalyse durchführen, ihr Nutzen ist jedoch nur partiell und verbessert den Gesamtprozess nicht. Hier sollte es zunächst von höherer Priorität sein, die generelle Funktionsfähigkeit und Zielerreichung zu sichern.

Bei fachlich sinnvoll strukturierten Prozessen kann die Kommunikationsanalyse sowohl auf statischer als auch auf dynamischer Ebene nutzbringend zur Effizienzsteigerung der Projektkommunikation eingesetzt werden. Dies wird die nachfolgende Diskussion zusammen mit der in Kapitel 4 präsentierten Validation zeigen.

Die Analyse lässt sich unterteilen in

- vorbereitende Schritte wie
  - der Fokussierung auf bestimmte Charakteristika für Teilbereiche einer Prozesslandschaft,
  - der Attributierung von Landschaftselementen
- und ausführende Schritte wie
  - dem eigentlichen Messen bzw. Berechnen von Attributwerten,
  - dem Bewerten der Kommunikationscharakteristika.

Beim zweiten Punkt muss man sich darüber im Klaren sein, dass bei jeder Messung aus der Vielzahl von Attributen eines Sachverhaltes eine einzelne abstrakte Zahl resultiert. Das Process Landscaping verwendet diese Abstraktion als eine – von Zeigler, Praehofer und Kim ausführlich diskutierte – Methode zur Reduzierung der Komplexität eines Modells „unter Beibehaltung seiner Gültigkeit innerhalb eines Experimentierrahmens“ [ZPK00] und sieht sie damit als eine gültige Modellvereinfachung an. Wie die Abstraktion von zu analysierenden Charakteristika einer Prozesslandschaft zu Attributen von Landschaftselementen durchgeführt wird, ist jeweils in Abhängigkeit der konkret betrachteten Eigenschaften erläutert. Die Bewertung der Kommunikationscharakteristika erfolgt anschließend auf Grundlage der gemessenen Attributwerte zusammen mit den sich daraus ergebenden Erkenntnissen über die Struktur einer Prozesslandschaft.

Die Analyseschritte werden unabhängig von den im Einzelnen zu analysierenden Eigenschaften immer in einer festen Reihenfolge durchlaufen, und das Vorgehen bei der Konzentration auf Teilmengen von Landschaftselementen weist für alle betrachteten Eigenschaften eine identische Struktur auf. Entsprechend ist dieses Unterkapitel gegliedert: Abschnitt 2.3.1 diskutiert zunächst die Erstellung unterschiedlicher Sichten auf eine Prozesslandschaft für eine Fokussierung auf bestimmte zu untersuchende Charakteristika. Abschnitt 2.3.2 beschreibt anschließend diejenigen Attribute, die statische Kommunikationseigenschaften ausmachen. In Abschnitt 2.3.3 finden sich die für eine Analyse dynamischer Kommunikationseigenschaften erforderlichen Attribute. Nach der Diskussion dieser vorbereitenden Schritte erfolgt in den Abschnitten 2.3.4 bis 2.3.5, die die Bemessungsgrundlagen und Bewertungsprinzipien der verschiedenen Eigenschaften erläutern, die Diskussion der ausführenden Schritte.

2.3.1 Umstrukturierung zur Erstellung unterschiedlicher Sichten auf eine Prozesslandschaft

Die in Abschnitt 2.2 vorgestellten Modellierungsschritte des Process Landscaping haben eine Prozesslandschaft zum Ergebnis, die entlang eines Kundenlebenszyklus entworfen wurde und somit insbesondere kausale Zusammenhänge der modellierten Aktivitäten widerspiegelt. Die entstandene Form der hierarchisch aufgebauten Struktur

wird im Rahmen des Process Landscaping als logische Sicht auf eine Prozesslandschaft bezeichnet (vgl. Abschnitt 1.3). Verteilungsaspekte sind innerhalb dieser Sicht durch Lokationsattribute berücksichtigt, die für jede Aktivität den Standort festlegen, an dem diese ausgeführt werden. Sie sind jedoch in der grafischen Repräsentation einer Prozesslandschaft nicht immer deutlich erkennbar. Dies erschwert eine Analyse mit Blick auf eben diese Verteilungsaspekte.

Um lokale Verteilungsaspekte innerhalb der logischen Sicht hervorzuheben, bedarf es einer Möglichkeit, die gegebene Prozesslandschaft nach anderen Ordnungskriterien – eben der räumlichen Verteilung von Aktivitäten – umzustrukturieren, ohne dass dabei bereits modellierte Informationen verloren gehen. Abbildung 2.6 verdeutlicht diese Situation zunächst an einem abstrakten Beispiel. Dabei wird der Fokus auf die Auswirkungen einer Umstrukturierung auf die betroffenen Aktivitäten gelegt. Von der Darstellung von Koordinationsinformationen für die Weiterleitung von Nachrichten und Objekten ist hier abstrahiert worden, da diese unter anderem von der Notation einer Prozesslandschaft abhängt. Diese ist jedoch bislang noch nicht diskutiert worden; eine abstrakte Darstellung würde den Rahmen der Abbildung sprengen und kaum zum besseren Verständnis des Umstrukturierungskonzeptes beitragen.

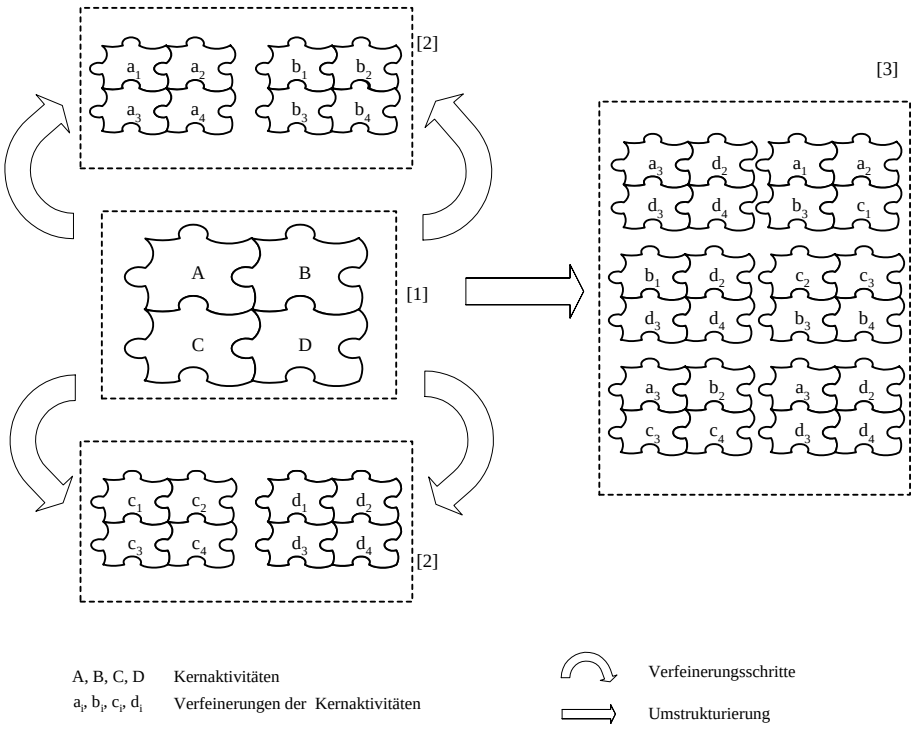


Abbildung 2.6: Umstrukturierung einer Prozesslandschaft

Der in Abbildung 2.6 mit [1] gekennzeichnete Ausschnitt repräsentiert die oberste Ebene einer beliebigen Prozesslandschaft, die aus den vier Kernaktivitäten *A*, *B*, *C* und *D* besteht. Diese sind als relativ grobe, zusammengehörende Puzzleteile dargestellt. Grenzen ihre Ränder an ein anderes Puzzleteil, so wird dies als existierende Schnittstelle interpretiert. Die beiden mit [2] gekennzeichneten Ausschnitte deuten an, dass die Kernaktivitäten jeweils durch eine Menge von weiteren Aktivitäten verfeinert worden sind. Die Zugehörigkeit zu ihrer jeweiligen Kernaktivität wird durch die Verwendung gleicher Buchstaben deutlich, wobei die Kernaktivitäten mit einem Großbuchstaben versehen sind, die verfeinernden Aktivitäten jeweils mit dem entsprechenden Kleinbuchstaben. Für die Erstellung der lokalen Sicht muss in einem ersten Schritt sichergestellt werden, dass für jedes Element der Prozesslandschaft – zumindest für alle Aktivitäten – der Ort, an dem diese ausgeführt werden, angegeben ist. Die Umstrukturierung kann dann anhand der verschiedenen angegebenen Orte erfolgen.

Der mit [3] gekennzeichnete Ausschnitt skizziert die lokale Sicht der so verfeinerten Prozesslandschaft. Hier wird deutlich, dass Umstrukturierung nicht nur eine einfache Neugruppierung bestehender Aktivitäten bedeutet, da einige von ihnen durchaus an mehreren Orten vorhanden sein können und deshalb in der lokalen Sicht mehrfach dargestellt sein müssen. In Abbildung 2.6 sind daher im rechten Ausschnitt unter anderem die Aktivitäten a_3, d_2, d_3 und d_4 mehrfach vorhanden. Die verschiedenen Lokalitäten innerhalb der Prozesslandschaft sind durch die jeweils voneinander abgegrenzt dargestellten Puzzleteile angedeutet. Schnittstellen zwischen ihnen sind noch nicht klar erkennbar.

Für eine analytische Betrachtung der Kommunikation muss auch die Koordination der Informationsverteilung auf verschiedene Standorte explizit festgelegt werden. Konkret bedeutet dies, dass eine Aktivität die von ihr bereitzustellenden Daten nicht nur produziert, sondern auch entscheidet, welche der nachfolgenden Aktivitäten bzw. welcher Standort diese Daten erhalten soll. Teilweise benötigen alle nachfolgenden Aktivitäten die erzeugten Daten, manchmal sind sie nur für eine Teilmenge der nachfolgenden Aktivitäten bestimmt und damit nur für einen bzw. einige Standorte. Bezogen auf das abstrakte Beispiel in Abbildung 2.6 muss beispielsweise festgelegt werden, ob Informationen, die in der logischen Sicht von Aktivität a_2 an Aktivität a_3 weitergeleitet wurden, in der lokalen Sicht an alle drei existierenden Aktivitäten a_3 weiterzuleiten sind. Stellt diese abstrakte Prozesslandschaft die Beauftragung der Entwicklung einer einzelnen Komponente im Rahmen einer komponentenbasierten Softwareentwicklung dar, so ist der Auftrag sicherlich nur an eine der drei Aktivitäten a_3 zu vergeben. Wird aber hier die Weiterleitung von Richtlinien seitens des Qualitätsmanagements an die verschiedenen Komponentenentwicklungsstandorte dargestellt, sind diese an alle drei Standorte zu versenden, und es entstehen weitere Schnittstellen, die in der logischen Sicht nicht erkennbar sind. Für jede Situation muss also die Informationsverteilung kontextabhängig vom Modellierer im Modell abgebildet werden.

Die Hervorhebung von Verteilungsaspekten innerhalb einer Prozesslandschaft durch die Erstellung einer lokalen Sicht produziert zwar eine Menge zusätzlicher Schnittstellen im Modell, erschwert damit jedoch nicht gleichzeitig das Verständnis für die

Prozesslandschaft. Sie lenkt von logischen Abhängigkeiten der Aktivitäten ab und erleichtert somit die Konzentration auf Verteilungsaspekte. Dies ist für alle diskutierten Analysen des Process Landscaping zumindest in Teilbereichen erforderlich.

2.3.2 Attributierung einer Prozesslandschaft zur Analyse statischer Kommunikationseigenschaften

Die Analyse statischer Kommunikationseigenschaften schließt die Betrachtung von Abläufen und Häufigkeiten ihrer Durchführung explizit aus und beschränkt sich auf die Kommunikationsinfrastruktur einer verteilten Prozesslandschaft. Diese lässt sich durch Kommunikationskanäle und deren Eigenschaften beschreiben. Unter einem *Kommunikationskanal* ist dabei ein Paar von Aktivitäten zu verstehen, das einen Informationstyp in einer festgelegten Richtung austauscht. Insgesamt sind also immer drei Elemente einer Prozesslandschaft und eine definierte Richtung des Informationsaustausches notwendig, um einen Kommunikationskanal zu bilden. Die Menge aller Kommunikationskanäle repräsentiert die Kommunikationsinfrastruktur einer Prozesslandschaft. Verbindet ein Kommunikationskanal dazu noch Aktivitäten, die an unterschiedlichen Standorten stattfinden, können über Eigenschaften der Kommunikationsinfrastruktur gleichzeitig auch Aussagen über die Verteilungsstruktur der Prozesslandschaft gemacht werden. Das Process Landscaping abstrahiert Eigenschaften dieser Infrastruktur über die Attribute *Synchronität*, *Verschlüsselung*, *Privatheit*, *Veränderbarkeit*, *Mehrfachversand* und *Persistenz*. Diese *Kommunikationsattribute* werden im Folgenden näher erläutert.

Während die ersten fünf Attribute bestimmen, wie eine Information ausgetauscht wird, legt das Attribut *Persistenz* fest, ob eine Information lokal gespeichert wird, bevor sie versandt wird bzw. nachdem sie empfangen worden ist. Ein synchroner Kommunikationskanal schließt Datenträger wie Briefe oder Faxe aus, während ein asynchroner Kanal diese durchaus repräsentieren kann.

Verschlüsselter Informationsaustausch erhöht die Komplexität der Kommunikationsinfrastruktur, da dies Kodierungswerkzeuge beim Sender und Dekodierungswerkzeuge beim Empfänger voraussetzt. Ein privater Informationsaustausch hat ebenfalls Auswirkungen auf die Komplexität, da er erhöhte Sicherheitsanforderungen an die Infrastruktur stellt. Hierfür muss gewährleistet werden, dass eine Information ausschließlich vom angegebenen Empfänger lesbar ist. Ist dieses Attribut gesetzt, wird gleichzeitig die Identifikation besonders schützenswerter Informationen unterstützt. Die Verschlüsselung einer Information erhöht somit die Sicherheit während des Datenaustausches, während die Privatheit den Empfängerkreis für sicherheitssensible Daten einschränkt.

Das Attribut *Veränderbarkeit* setzt voraus, dass Informationen nicht schreibgeschützt in einem Datenaustauschformat versendet werden, das dem Empfänger direkte Änderungen an den erhaltenen Informationen erlaubt: Eine Postscript-Datei oder ein signiertes Dokument weisen beispielsweise eine nicht veränderbare Form auf. Das Attribut *Mehrfachversand* hat lediglich Auswirkung auf die Infrastruktur der sendenden Aktivität, da hier entsprechende Funktionalitäten zur Verfügung stehen müssen: Brie-

fe müssen beispielsweise mehrfach gedruckt werden, Mail-Werkzeuge benötigen eine entsprechende Adressenverwaltung.

Eine Kommunikation kann trotz korrekter Definition eines Kommunikationskanals immer noch daran scheitern, dass z.B. die sendende Aktivität eine Information verschlüsselt, der Empfänger jedoch eine unverschlüsselte Nachricht erwartet und seine Infrastruktur daher keine Dekodierungsfunktionalität zur Verfügung stellt. Ist die Ausprägung der Attribute *Synchronität*, *Verschlüsselung*, *Privatheit*, *Veränderbarkeit* und *Mehrfachversand* jeweils für beide beteiligten Aktivitäten eines Kommunikationskanals gleich, wird er im Folgenden als *funktionsfähig* bezeichnet. Das Persistenzattribut spielt hierbei eine untergeordnete Rolle, da es sich nicht auf die Art einer Kommunikation auswirkt. Damit ist die Beschreibung einer komplexen Kommunikationsinfrastruktur auf eine kleine Menge von Attributen mit jeweils einem Minimum an Ausprägungsmöglichkeiten (*ja* bzw. *nein*) reduziert.

Ein weiteres Attribut, welches zumindest einen indirekten Einfluss auf die statischen Kommunikationseigenschaften einer Prozesslandschaft hat, ist das *Lokationsattribut*. Wie bereits beschrieben, legt es für jede Aktivität den Standort fest, an dem sie ausgeführt wird. Dies erlaubt eine Kommunikationsanalyse sowohl an einem als auch zwischen verschiedenen Orten. Insbesondere für die Analyse von verteilter Kommunikation ist das Lokationsattribut von großer Wichtigkeit, denn immer dort, wo Kommunikation über Ortsgrenzen hinaus stattfindet, erhöht sich gemäß der in Abschnitt 2.1 diskutierten Annahme, dass externe Kommunikation im Vergleich zu interner als teurer zu bewerten ist, auch der entsprechende Kostenfaktor.

Für eine sinnvolle Analyse sind in Abhängigkeit der zu untersuchenden Eigenschaften noch einige Konsistenzprüfungen durchzuführen. Bei der Analyse statischer Kommunikationseigenschaften betrifft dies insbesondere die Funktionsfähigkeit von Kommunikationskanälen. Das Identifizieren nicht funktionsfähiger Kommunikationskanäle deckt Abstimmungs- und Modellierungsfehler auf. Ein Abstimmungsfehler liegt beispielsweise vor, wenn die einem Kommunikationskanal zugehörigen Aktivitäten von verschiedenen Modellierern der Prozesslandschaft hinzugefügt wurden und deren jeweilige Interviewpartner unterschiedliche Vorstellungen über ihre Kommunikationsinfrastruktur haben. Derartige Unstimmigkeiten können nach ihrer Identifikation schnell behoben werden.

Weitere Konsistenzprüfungen betreffen den Abgleich der Ausprägungen verschiedener Kommunikationsattribute. So darf ein als nicht veränderbar verschicktes Dokument von der Empfängeraktivität nicht überschrieben werden. Ist in der Prozesslandschaft trotzdem ein Schreibzugriff auf dieses Dokument dargestellt, so ist vom Modellierer zu überprüfen, ob diese Situation korrekt ist. Es ist möglich, dass die Empfängeraktivität das Dokument zu einem späteren Zeitpunkt bearbeiten darf, was aber aufgrund der fehlenden Ablaufinformation im Modell nicht unbedingt deutlich wird.

Damit sind alle für die Analyse statischer Kommunikationseigenschaften als sinnvoll erachteten Attribute eingeführt und ihre Bedeutung – zunächst losgelöst von einer konkreten Beispiellandschaft – erläutert worden. In Abschnitt 2.3.4 wird ein Interpretationsrahmen aufgespannt, der eine für den Anwender nutzbringende Analyse dieser

Kommunikationseigenschaften erlaubt. Dabei wird insbesondere der Verteilungsaspekt innerhalb einer Prozesslandschaft berücksichtigt, die dazu in ihrer lokalen Sicht betrachtet wird. Eine Nutzenbetrachtung der Analyse für ein konkretes Beispiel erfolgt in Abschnitt 4.1.3.

2.3.3 Attributierung einer Prozesslandschaft zur Analyse dynamischer Kommunikationseigenschaften

Im Unterschied zur Analyse einer statischen Kommunikationsinfrastruktur schließt das Process Landscaping für die Analyse dynamischer Kommunikationseigenschaften die Berücksichtigung von Ablaufinformationen innerhalb einer Prozesslandschaft explizit ein. Damit greift hier die von Feiler und Humphrey aufgestellte Definition der Analyse (von Softwareprozessen) als „the use of process traces, process definitions, and process plans to assess process behaviour“ [FH93], die diejenigen Kommunikationseigenschaften betrachtet, die sich aus dem dynamischen Verhalten der Prozesslandschaft ergeben.

Im Vordergrund dieses Analyseschrittes stehen verschiedene Effizienzbetrachtungen. Sie basieren auf der simulativen Analyse verschiedener Landschaftseigenschaften, die wiederum über die Messung und Auswertung ihrer zugeordneten Attributwerte erfolgt. Dabei sollte zu Beginn der Analyse darauf geachtet werden, dass die zu untersuchende Prozesslandschaft ein möglichst homogenes Abstraktionsniveau aufweist. Nur so kann sichergestellt werden, dass die in einer Simulation gewonnenen Werte sinnvoll analysierbar sind. Des Weiteren ist zu beachten, dass die Simulationsergebnisse nicht global zu bewerten sind, sondern immer im Zusammenhang mit der konkret betrachteten Prozesslandschaft zu sehen sind. Beispielsweise müssen die Kommunikationskosten eines Standortes innerhalb einer Prozesslandschaft immer im Verhältnis zu den Kommunikationskosten anderer Standorte der Landschaft und im Verhältnis zu den Kommunikationskosten der gesamten Landschaft betrachtet werden, um qualitative Aussagen über die Höhe der Kosten treffen zu können. Ohne die Vergleichswerte anderer Standorte und ohne Kenntnisse über die Gesamtkosten bleiben qualitative Aussagen immer vage und sind damit nicht sinnvoll nutzbar.

Die Effizienzbetrachtungen berücksichtigen den in Abschnitt 2.1 formulierten Grundsatz der effizienten Kommunikation, indem sie die folgenden, die Kommunikationseffizienz beeinflussenden, Kriterien beleuchten:

- *Auslastungsgrad* verteilter Aktivitäten bzgl. der Kommunikationshäufigkeit: Hier liegt das Hauptaugenmerk auf der Leistungsfähigkeit der Aktivitäten in Abhängigkeit ihrer Auslastung. Dabei ist insbesondere darauf zu achten, dass Engpässe – also eine zu hohe Auslastung von Aktivitäten – auch die Effizienz anderer Aktivitäten beeinflussen, wenn nämlich letztgenannte von den Ergebnissen der ausgelasteten Aktivitäten abhängig sind.

- *räumliche Verteilung* von Aktivitäten bzgl. entstehender Kommunikationskosten: Die Aktivitäten einer Prozesslandschaft sollten so verteilt sein, dass die

Kosten, die durch Kommunikation zwischen Orten (externe Kommunikation) entstehen, möglichst gering sind.

- *Informationsverteilung* bzgl. der Zugriffshäufigkeit auf Informationen innerhalb einer Prozesslandschaft: Informationen, die häufig von Prozessteilnehmern angefragt werden, erfordern eine Kommunikationsstruktur, die die durch eine Beantwortung von Anfragen involvierten Aktivitäten unterstützen. Die betroffenen Aktivitäten können über ihre Rolle identifiziert werden, die sie bei einer initiierten Kommunikation spielen: Signalisiert ein Kommunikationspartner den Wunsch zu kommunizieren, übernimmt er die Rolle des Initiators. Sobald der zweite Kommunikationspartner auf diesen Wunsch reagiert, übernimmt er die Rolle des Responders; die Kommunikation kann beginnen. Diese gilt als erfolgreich, wenn ein Initiator ohne nennenswerte zeitliche Verzögerung die angefragte Information vom Responder erhält. Je gleichmäßiger die Rollen des Initiators und des Responders innerhalb einer Prozesslandschaft verteilt sind, desto höher ist die Anzahl erfolgreicher Kommunikationsversuche und desto effizienter damit die Informationsverteilung.

- *Informationsfluss* innerhalb einer Prozesslandschaft: Werden Daten gleichen Typs mehrfach zwischen verschiedenen Aktivitäten ausgetauscht und ist gleichzeitig der Bearbeitungsanteil innerhalb der Aktivitäten gering, so ist die Kommunikationseffizienz bzgl. des Informationsflusses als gering einzuschätzen. Die Minimierung dieser im Folgenden als Ping-Pong-Kommunikation [GW99b] bezeichneten Form des Informationsflusses hilft, die Kommunikationseffizienz innerhalb einer Prozesslandschaft zu steigern.

Die verschiedenen Aspekte der Kommunikationseffizienz lassen sich über Eigenschaften wie Kommunikationsaufkommen, zeitliche Prozessauslastung, durchschnittliche Weglänge für Informationsobjekte und die Häufigkeit von Nachfrage und Bereitstellung von Informationen charakterisieren, die wiederum anhand konkreter Attribute der Elemente einer Prozesslandschaft gemessen und bewertet werden können. Für Letzteres wird in Abschnitt 2.3.5 ein geeigneter Interpretationsrahmen diskutiert, dessen Nutzen in Abschnitt 4.2.2 validiert wird.

Abbildung 2.7 stellt den Zusammenhang zwischen den verschiedenen Effizienzbetrachtungen, ihren charakteristischen Eigenschaften und den sie beschreibenden Attributen dar.

Die *Auslastung* einer Aktivität lässt sich definieren als Verhältnis des Zeitraums, in dem diese durchgeführt wird, zur Gesamtzeit der Durchführung aller Aktivitäten einer Prozesslandschaft. Da dieser für die Durchführung benötigte Zeitraum von der Komplexität der zu verarbeitenden Daten abhängt, lässt sich die Kommunikationseffizienz bzgl. des Auslastungsgrades mit Hilfe eines Komplexitätswertes realitätsnäher messen, der den Zeitaufwand einer Aktivität für die Verarbeitung einer Information beeinflusst (vgl. Abbildung 2.7). Das entsprechende Attribut *Komplexitätswert* ist hier definiert als eine Zahl zwischen eins und zehn, mit dem eine für den durchschnittli-

chen Zeitaufwand angegebene Verarbeitungsdauer multipliziert wird (vgl. hierzu auch Abschnitt 4.2.1, Seite 4.2.1).

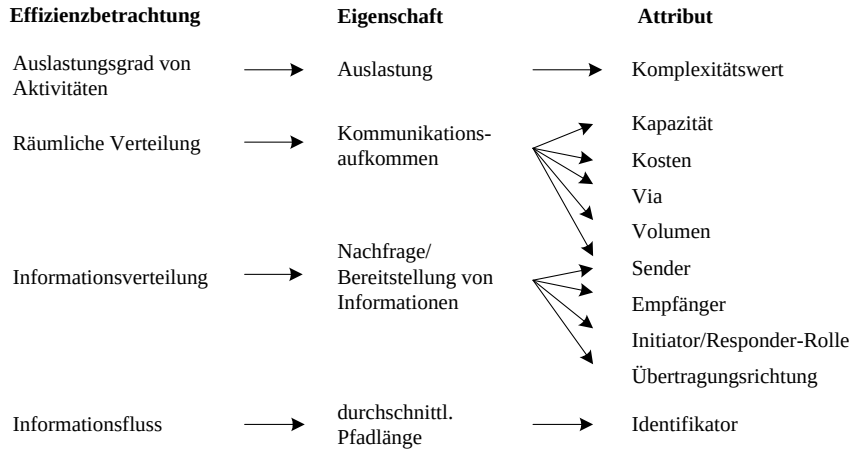


Abbildung 2.7: Zusammenhang zwischen Effizienzbetrachtungen, Eigenschaften und Attributen einer Prozesslandschaft

Die Kommunikationseffizienz bzgl. der räumlichen Verteilung von Aktivitäten lässt sich durch die Eigenschaft *Kommunikationsaufkommen* beschreiben. Diese dient als Bemessungsgrundlage für den auf einen konkreten Zeitraum bezogenen und in Kosten umgerechneten Aufwand von Aktivitäten, der durch Kommunikation entsteht. Die Berechnung des Kommunikationsaufkommens erfolgt über Attribute beteiligter Kommunikationskanäle und auszutauschender Informationen. Für eine Menge von Kommunikationskanälen sind hierfür jeweils ihre Kapazität und ein Kostenmaß als gemeinsame Attribute festzulegen; über die auszutauschenden Informationen müssen jeweils ihr Sender, das Volumen und der sie übertragende Kommunikationskanal bekannt sein.

Bei einem Informationsaustausch übernimmt die anfragende Aktivität die Rolle des Initiators und die antwortende Aktivität die des Responders. Diese Rollen sind unterschiedlich zu denen eines Senders bzw. Empfängers. So ist etwa bei einem Informationsabruf von einer Webseite der Initiator der Kommunikation gleichzeitig auch der Empfänger der Daten. Je gleichmäßiger die Rollen Initiator und Responder innerhalb einer Prozesslandschaft verteilt sind, desto effizienter ist die betrachtete Informationsverteilung. Insgesamt müssen, wie in Abbildung 2.7 dargestellt, Empfänger und Sender als Attribute der auszutauschenden Informationen sowie die Initiator-/Responder-Rolle und die Übertragungsrichtung als Attribute der beteiligten Kommunikationskanäle gemessen werden, um Aussagen über die Kommunikationseffizienz bzgl. einer Informationsverteilung treffen zu können.

Die *durchschnittliche Pfadlänge*, die eine Information innerhalb einer Prozesslandschaft bis zur ihrer endgültigen Verarbeitung benötigt hilft, weitere lohnenswerte Teil-

bereiche dieser Landschaft als Analysekandidaten zur möglichen Effizienzsteigerung aufzuspüren. Diese durchschnittliche Pfadlänge ergibt sich aus der Menge der Relationen zwischen Aktivitäten und Schnittstellen, die ein Informationsobjekt, von einer verfeinerten Aktivität ausgehend, passiert, bevor es zurück zu eben dieser verfeinerten Aktivität gesendet wird. Diese an sich statische Länge wird während einer Simulation mehrfach gemessen, wobei sich durch die möglichen unterschiedlichen Pfade, die ein beobachtetes Informationsobjekt durchlaufen kann, die gesuchte durchschnittliche Pfadlänge berechnen lässt. Voraussetzung für die Messung einer solchen Pfadlänge ist eine eindeutige Identifizierungsmöglichkeit der betrachteten Informationen auf dem Weg durch die Prozesslandschaft und damit die Existenz eines entsprechenden Identifikator-Attributs für jede zu betrachtende Information (vgl. Abbildung 2.7).

Bei allen Arten der Effizienzbetrachtung liegt das Hauptaugenmerk auf verschiedenen Verteilungsaspekten innerhalb einer gegebenen Prozesslandschaft: Beim Auslastungsgrad liegt der Fokus auf einer gleichmäßigen Arbeitsverteilung, bei der räumlichen Verteilung auf der lokalen Verteilung der Aktivitäten selbst, bei der Informationsverteilung auf einer gleichmäßigen Verteilung der Initiator- und Responder-Rolle und beim Informationsfluss auf einem möglichst gleichverteilten Bearbeitungsanteil von Aktivitäten zur Verarbeitung konkreter Informationen. Wann eine Verteilung als effizient eingestuft werden kann und welche Interpretationsprinzipien dabei zu berücksichtigen sind, wird in Abschnitt 2.3.5 diskutiert.

Abschließend ist noch festzuhalten, dass auch die Analyse dynamischer Kommunikationseigenschaften einige Konsistenzprüfungen voraussetzt. Diese bestehen jedoch nicht nur darin, die Vollständigkeit einer Attributierung zu überprüfen, sondern auch die Kongruenz weiterer Attributwerte zu sichern. Beispielsweise müssen für Sender und Empfänger einer Nachricht jeweils unterschiedliche Aktivitäten angegeben sein. Diese müssen wiederum mit Kommunikationskanälen verbunden sein, die innerhalb der betrachteten Teilmenge aller existierenden Kommunikationskanäle liegen. Anders ausgedrückt muss sichergestellt sein, dass ausgewählte Teilmengen einer Prozesslandschaft alle Landschaftselemente enthalten, die für eine konkrete Analyse zu berücksichtigen sind.

Nachdem mit der Umstrukturierung einer logischen in die lokale Sicht einer Prozesslandschaft und ihrer Attributierung alle vorbereitenden Schritte für die Analysen des Process Landscaping erläutert worden sind, werden im Folgenden die Interpretationsrahmen der verschiedenen betrachteten Eigenschaften diskutiert.

2.3.4 Interpretation statischer Kommunikationseigenschaften

Das Process Landscaping unterstützt bei der Analyse statischer Kommunikationseigenschaften vor allem die oberen Ebenen einer Prozesslandschaft, innerhalb derer Ablaufinformationen noch nicht modelliert sind (vgl. Abschnitt 2.2.3). Der Schwerpunkt der Analyse liegt auf der Kommunikationsinfrastruktur. Für diese Analyse wird die Vielfalt der Werte der Kommunikationsattribute wie folgt bewertet: Mit zunehmender Vielfalt der Werte einer betrachteten Menge von Kommunikationsattributen steigt

die *Komplexität* der betrachteten Infrastruktur. Die *Dichte* einer Kommunikationsinfrastruktur wird über die Anzahl existierender Kommunikationskanäle innerhalb einer betrachteten Teilmenge einer Prozesslandschaft berechnet. Sie ist unabhängig von der Ausprägung der Kommunikationsattribute und damit unabhängig von der zuvor beschriebenen Komplexität einer Infrastruktur.

Die Interpretation der Eigenschaften *Komplexität* und *Dichte* einer Kommunikationsinfrastruktur ist an die Begriffe *Kopplung* und *Kohäsion* aus dem Bereich der komponentenbasierten Softwareentwicklung angelehnt. Dort wird angestrebt, die einzelnen Komponenten eines Softwaresystems möglichst mit hoher Kohäsion und geringer Kopplung zu entwickeln. Hier dienen sie als Maß bei der Zusammenfassung einzelner Aktivitäten bzw. einer Menge von Aktivitäten an einem gemeinsamen Standort und bei der Aufteilung auf verschiedene Standorte. Unter dem Oberbegriff der *Kopplung von Standorten* betrachtet das Process Landscaping die Kommunikation zwischen Aktivitäten verschiedener Standorte. Die *Kohäsion eines Standortes* beinhaltet entsprechend die Kommunikation zwischen Aktivitäten eines gemeinsamen Standortes. Je höher die Kohäsion und je geringer die Kopplung von Standorten, desto besser wird die Zuordnung der Aktivitäten einer Prozesslandschaft zu diesen Standorten bewertet.

Der Interpretationsrahmen zur Analyse statischer Kommunikationseigenschaften einer Prozesslandschaft kann nun wie folgt aufgespannt werden: Für Teilbereiche einer Prozesslandschaft untersucht das Process Landscaping die

- Kommunikation zwischen verschiedenen Orten mit Hilfe der
  - *Kopplungsdichte*,
    die sich aus der Anzahl der Kommunikationskanäle zwischen zwei ausgewählten Standorten oder bezüglich eines einzelnen zu mehreren anderen Standorten ergibt.
  - *Kopplungskomplexität*,
    die sich aus der Anzahl unterschiedlich ausgeprägter Kommunikationskanäle zwischen zwei ausgewählten Standorten oder bezüglich eines einzelnen zu mehreren anderen Standorten ergibt.
  - *Kopplungszahl*,
    die sich aus der Anzahl der Kanäle ergibt, die Aktivitäten an einem ausgewählten Standort mit Aktivitäten an anderen Standorten verbinden.
- und die Kommunikation innerhalb eines Ortes mit Hilfe der
  - *Kohäsionsdichte*,
    die sich aus der Anzahl der Kommunikationskanäle an genau einem Standort ergibt.
  - *Kohäsionskomplexität*,
    die sich aus der Anzahl der vorkommenden Kombinationen unterschiedlich ausgeprägter Kommunikationsattribute an genau einem Standort ergibt.

– *Kohäsionszahl*,
die sich aus der Anzahl der Aktivitäten ergibt, die innerhalb eines ausgewählten Standortes durch Kommunikationskanäle verbunden sind.

Ziel der Analyse von Kopplung und Kohäsion mit Hilfe der obigen Begriffe ist es, bezüglich verschiedener Umstrukturierungsziele der Kommunikationsinfrastruktur einer Prozesslandschaft die jeweils bestmögliche Alternative herauszufinden. So kann etwa bei einem geplanten Auflösen des Standortes einer Firma die Frage geklärt werden, wie die einzelnen Aktivitäten dieses Standortes auf bestehende oder neue Standorte verteilt werden sollten. Als Bemessungsgrundlage dient hierbei die Kopplung zu Aktivitäten an anderen Standorten und die Kohäsion zu anderen Aktivitäten am gleichen Standort. Eine Umverteilung der Aktivitäten des aufzulösenden Standortes ist mit dem Ziel zu untersuchen, Orte mit hoher Kohäsion und geringer Kopplung zu erhalten. Dabei folgt man der Annahme, dass Kommunikation zwischen zwei Orten teurer ist als die Kommunikation zwischen Aktivitäten, die am selben Ort ausgeführt werden (vgl. Abschnitt 2.1). Es ist jedoch nicht sinnvoll, Kommunikation aus Gründen der Kostenersparnis ersatzlos zu streichen, sondern möglichst durch Verlegung von Aktivitäten teure externe Kommunikation durch günstigere interne Kommunikation zu ersetzen.

Die Bemessungsgrundlage für die Kommunikationskosten ergibt sich aufgrund der bislang betrachteten statischen Kommunikationsinfrastruktur wie folgt: Zunächst wird das Verhältnis zwischen der externen und internen Kommunikationsdichte eines Ortes *o* als Quotient betrachtet. Da auf den oberen Ebenen einer Prozesslandschaft noch keine Ablaufinformationen berücksichtigt sind, wird die Nutzungshäufigkeit jedes Kommunikationskanals abstrakt auf einer Skala von eins bis zehn festgelegt. Diese Skala hat sich in verschiedenen Interviews als sinnvoll für die Abschätzung von Nutzungshäufigkeiten der Kommunikationskanäle herausgestellt. Die Abschätzung sollte mit Hilfe der Prozessbeteiligten bzw. -verantwortlichen durchgeführt werden, die von entsprechenden Erfahrungswerten abstrahieren können.

Die Kosten pro Nutzung eines Kommunikationskanals werden auf einer Skala von eins bis fünf bestimmt. Diese Skala ergibt sich aus der Anzahl der zu berücksichtigenden Kommunikationsattribute. Die Werte der Kopplungs- und Kohäsionskomplexitäten dienen hierbei als Richtwerte: Je komplexer die Kommunikationsinfrastruktur zwischen zwei Aktivitäten oder zwischen zwei Orten, desto höher der Wert auf der Kostenskala. Insgesamt lässt sich der so aufgebaute Kommunikationskostenfaktor Kkf berechnen als

$$\frac{Kopplungsdichte(o) \times (Nutzungsh\ddot{a}ufigkeit \times Kosten)\ pro\ externer\ Kanal}{Koh\ddot{a}sionsdichte(o) \times (Nutzungsh\ddot{a}ufigkeit \times Kosten)\ pro\ interner\ Kanal}$$

Er ermöglicht es, auf einer einfachen Bemessungsgrundlage für die statischen Kommunikationsaspekte einer Prozesslandschaft zu bleiben. Je kleiner der Kkf eines Standortes ist, desto effizienter ist seine Kommunikationsinfrastruktur.

Wird die Minimierung dieses Kkf ausschließlich und konsequent als Optimierungsansatz verwendet, führt dies zu dem Ergebnis, dass alle Aktivitäten am gleichen Ort stattfinden. In der Realität ist dieser ausschließliche Ansatz jedoch unbrauchbar, da

viele Aktivitäten aus unterschiedlichsten Gründen über verschiedene Standorte verteilt stattfinden. Kundennähe, Standortkosten oder gemeinsam mit anderen zu nutzende Ressourcen sind nur einige dieser Gründe. Eine Grundvoraussetzung für Optimierungsansätze sollten daher vorhandene Vorüberlegungen bzw. Strategien sein, wie und wo eine Prozesslandschaft verändert werden soll. Denkbare Strategien in diesem Zusammenhang sind

- die geplante Auflösung eines Standortes,
- das geplante Zusammenlegen von Standorten,
- die geplante Verlagerung einzelner Aktivitäten oder
- der geplante Aufbau eines neuen Standortes.

Grundsätzlich sind alle vier Strategien zur Veränderung bzw. Verbesserung einer Prozesslandschaft ähnlich strukturiert. Sie erfordern jedoch unterschiedliche Vorgehensweisen bei der Optimierung. Soll beispielsweise ein Standort aufgelöst werden, so ist möglichst ein Ort mit hohem Kkf in Betracht zu ziehen. Werden seine Aktivitäten auf diejenigen Standorte verteilt, mit denen sie zuvor intensiv extern kommuniziert haben, kann diese Kommunikation nach ihrer Verlagerung durch günstigere interne Kommunikation ersetzt werden. Sollen Standorte zusammengelegt werden, so sind zwei Orte auszuwählen, die eine relativ hohe Kopplungsdichte aufweisen, damit der Kkf des resultierenden Ortes möglichst gering gehalten werden kann. Nach der Verlagerung einer Aktivität zu einem anderen Standort sollte der Kkf beider Orte (Ursprung und Ziel) mindestens gleich bleiben, jedoch nicht größer als vorher sein. Dadurch wird verhindert, dass eine Aktivität aus einem Verbund mit hoher Kohäsion gelöst wird. Für den geplanten Aufbau eines neuen Standortes durch Verlagerung verschiedener Aktivitäten gilt schließlich, dass die Summe aller Kommunikationskostenfaktoren (inklusive des neuen Standortes) möglichst kleiner werden sollte.

Schließlich bleibt noch festzuhalten, dass sich alle vier Strategien ausschließlich auf statische Eigenschaften der Kommunikationsinfrastruktur einer Prozesslandschaft stützen. Diese Form der Analyse eignet sich daher besonders für unstrukturierte und flexible Prozesse ohne festgeschriebene Abläufe wie beispielsweise Softwareprozesse mit einem hohen Maß an Kreativität und Flexibilität. Die hier diskutierten Optimierungsansätze sind jedoch nicht als einzig mögliche Kriterien zur Optimierung einer Prozesslandschaft anzusehen. So werden oft politische Überlegungen eingesetzt, wenn es um die Veränderung von Standorten eines Unternehmens geht. Die hier beschriebenen Strategien sind als weitere mögliche Entscheidungshilfen für Prozessverantwortliche gedacht, wie sie bislang in der Literatur noch nicht berücksichtigt worden sind.

Die vier gerade diskutierten Optimierungsansätze werden in Abschnitt 4.1.2.2 auf eine Beispielprozesslandschaft angewendet und ihr Nutzen validiert.

2.3.5 Interpretation dynamischer Kommunikationseigenschaften

Die Interpretation dynamischer Kommunikationseigenschaften einer verteilten Prozesslandschaft muss vor allem das dynamische Verhalten beteiligter Aktivitäten während der Kommunikation berücksichtigen. Damit reichen statische Messungen von Attributwerten nicht mehr aus. Das Process Landscaping bietet die simulative Analyse für die Auswertung des dynamischen Verhaltens an.

Für eine Nutzenbetrachtung und damit auch für eine qualitative Bewertung der durch die Simulation einer Prozesslandschaft erhaltenen Zahlengrößen verschiedener Effizienzaspekte stellt sich die Frage, wie eine Einschätzung der Qualität dieser Effizienzaspekte vorgenommen werden kann. Beim Process Landscaping wird dies in Anlehnung an die AMI-Methode realisiert [DF96]. Die Methode besteht aus zwölf Schritten, die zu vier Aktivitäten (Assess, Analysis, Metricate, Improve) zusammengefasst sind. Die Aktivität *Assess* definiert nach der ersten Einschätzung einer Prozesslandschaft Primärziele wie z.B. die Verbesserung der Kommunikationseffizienz des modellierten Entwicklungsprozesses. Diese Primärziele werden nach ihrer Validation durch die Aktivität *Analyse* in Teilziele wie z.B. der Kommunikationseffizienz bzgl. einer räumlichen Verteilung miteinander kommunizierender Aktivitäten und der Kommunikationseffizienz bzgl. des Informationsflusses innerhalb einer Prozesslandschaft zerlegt, für die dann Metriken identifiziert werden können. Die Aktivität *Metricate* ist für das Sammeln der für eine Analyse notwendigen Daten zuständig (im Rahmen des Process Landscaping per Simulation). Schließlich beinhaltet die Aktivität *Improve* die Auswertung der Daten, wobei sie die quantitativen Analyseergebnisse auch qualitativ bewertet.

Man kann jedoch auch – wie bei der Analyse statischer Kommunikationseigenschaften – ein konkretes Optimierungsziel voraussetzen und auf der Basis einer gegebenen Attributausprägung (Ist-Zustand einer gegebenen Prozesslandschaft) die effizienteste Alternative für eine geplante Attributausprägung (Soll-Zustand einer Prozesslandschaft) ermitteln. Die in Abschnitt 2.3.2 formulierten Optimierungsansätze, die eine Verlagerung von Aktivitäten innerhalb einer Prozesslandschaft auf andere Standorte implizieren, sind auch für die Analyse dynamischer Kommunikationseigenschaften geeignet. Modellierungsseitig sollte zuvor – wie bereits in Abschnitt 2.3.3 erläutert – darauf geachtet werden, dass die betrachteten Teilbereiche einer gegebenen Prozesslandschaft in etwa die gleiche Abstraktionstiefe haben, um zu sinnvollen und damit verwertbaren Ergebnissen zu gelangen. Die Bewertung der Ergebnisse für diese Teilbereiche kann dann mit Hilfe von Vergleichswerten bezüglich anderer Teilbereiche erfolgen.

Im Folgenden wird auf die in Abschnitt 2.3.2 diskutierten Effizienzbetrachtungen eingegangen und jeweils erläutert, wie die Simulationsergebnisse hinsichtlich einer möglichen Effizienzoptimierung zu interpretieren sind.

Je gleichmäßiger die *Auslastung der Aktivitäten* innerhalb einer Prozesslandschaft, desto effizienter kann die Kommunikation innerhalb dieser Landschaft erfolgen. Als Bewertungsgrundlage sollte jedoch nicht nur die durchschnittliche Auslastung aller

Aktivitäten im Verhältnis zur Gesamtdauer einer Simulation dienen, da durch die Mittelwertbildung gerade die interessanten Extremwerte aus dem Blickfeld geraten. Die Betrachtung der Schwankungsbreite ist ebenfalls von großer Bedeutung, da ihr Auftreten zu zeitweiligen Verarbeitungsengpässen führen kann. Eine Auslastung von 100% über einen längeren Zeitraum hinweg zeigt meist eine überlastete Aktivität an. Zur Entlastung kann diese entweder von Teilaufgaben befreit oder mit mehr Ressourcen ausgestattet werden. Bei einem niedrigen Auslastungsgrad ist der Kontext näher zu betrachten. Es gibt durchaus Aktivitäten, die im Verlauf eines durchschnittlichen Kundenlebenszyklus nur selten aktiv werden. Aus dem Bereich der komponentenbasierten Softwareentwicklung sei hier die Abnahme eines erstellten Softwaresystems mit dem Kunden als Beispiel genannt. Hat die untersuchte Aktivität jedoch keinen solchen Ausnahmecharakter, so ist die geringe Auslastung in vielen Fällen auf eine Überlastung der ihr vorgelagerten Aktivitäten zurückzuführen.

Für die *räumliche Verteilung* miteinander kommunizierender Aktivitäten gilt: Je niedriger das externe Kommunikationsaufkommen (gemäß dem Prinzip, dass externe Kommunikation teurer ist als interne) und je gleichmäßiger dies innerhalb einer Prozesslandschaft verteilt ist, desto effizienter kann die Kommunikation durchgeführt werden. Als Bewertungsgrundlage wird das Verursacherprinzip herangezogen, bei dem einer sendenden Aktivität pro Informationsaustausch die bei der externen Kommunikation entstehenden Kosten berechnet werden. Eine Einzelberechnung erfolgt durch die einfache Formel:

$$\frac{Kapazität}{Volumen} \times Kostenmaß$$

Kapazität und Volumen werden hierbei in kB gemessen, für das Kostenmaß stehen verschiedene Kostenmodelle zur Verfügung, die in Anhang A.3 aufgeführt sind.

Gleichzeitig wird der Auslastungsgrad der an externer Kommunikation beteiligten Kommunikationskanäle berechnet. Ihre Nutzungsdauer im Verhältnis zur Gesamtzeit einer Simulation bestimmt das (externe) Kommunikationsaufkommen zwischen zwei Standorten. Als Faustregel für eine effiziente räumliche Verteilung sollte gelten, dass zwei Aktivitäten um so eher an einem gemeinsamen Standort auszuführen sind, je mehr sie miteinander kommunizieren.

Eine geschickte *Informationsverteilung* trägt – wie bereits in Abschnitt 2.3.3 erläutert – im Rahmen des Process Landscaping zu einer hohen Kommunikationseffizienz bei, wenn die Rollen Initiator und Responder und damit auch der Aufwand für die Reaktion auf Informationsanforderungen möglichst gleichmäßig auf alle Aktivitäten innerhalb einer Prozesslandschaft verteilt sind. Dies minimiert gleichzeitig den zeitlichen Aufwand einer anfragenden Aktivität, benötigte Informationen zu erhalten. Ein Aufwand kann in diesem Zusammenhang in zweifacher Hinsicht gemessen werden. Zum einen stellt jede Anfrage, die beantwortet werden muss, eine Unterbrechung der antwortenden Aktivität dar. Daher sollte die Anzahl von Anfragen pro Zeiteinheit ein bestimmtes Maß nicht überschreiten. Zum anderen entsteht ein Aufwand beim Responder dadurch, dass die angefragten Informationen bereitgestellt werden müssen.

Dieser Aufwand lässt sich als Zeit messen die der Responder braucht, um nach Bestätigung des Kommunikationswunsches die angefragten Informationen zu liefern. Der Aufwand für den Responder kann unter anderem dadurch reduziert werden, dass Informationen so weit wie möglich zwischengespeichert werden. Insbesondere wenn sich abzeichnet, dass ein und dieselbe Information von verschiedenen Anfragern angefordert wird, sollte über ein Auslagern der Informationsverteilung nachgedacht werden. Die Anfrage nach Richtlinien seitens verschiedener Komponentenentwicklungsstandorte an das Qualitätsmanagement innerhalb einer verteilten Softwareentwicklung ist ein Beispiel für eine solche Situation. Eine Verbesserung könnte dadurch erfolgen, dass die Richtlinien in einem Intranet abgelegt werden und dort jeder anfragenden Aktivität lesend zur Verfügung gestellt werden.

Die Kommunikationseffizienz bzgl. des *Informationsflusses* innerhalb einer Prozesslandschaft zeichnet sich durch eine möglichst große durchschnittliche Pfadlänge zwischen zwei verfeinerten Aktivitäten aus, die zur Verarbeitung einer konkreten Information erforderlich ist: Eine kurzer Pfad, auf dem ein Informationsobjekt, ausgehend von und endend in einer verfeinerten Aktivität, aber innerhalb einer zweiten verfeinerten Aktivität bearbeitet wird, weist auf eine geringe Bearbeitungstiefe hin. Eine solche Situation weist in aller Regel auf eine tayloristische Organisation hin [PW94], in der Prüfung und Bearbeitung einer Information strikt voneinander getrennt sind. Ist eine solche Situation jedoch nicht aufgrund einer tayloristischen Organisation entstanden, so sollte überlegt werden, ob die Aufgabenverteilung zugunsten einer höheren Bearbeitungstiefe verändert werden kann. Je kürzer also der Bearbeitungspfad einer Information, desto wahrscheinlicher ist es, einen ineffizienten Informationsfluss aufgespürt zu haben. Zwar ist die Länge eines Bearbeitungspfades eine statische Eigenschaft, die sich bereits durch die Modellierung der Prozesslandschaft ergibt. Während der Simulation werden jedoch – soweit vorhanden – verschiedene mögliche Pfade unterschiedlich häufig durchlaufen. So bestimmt die Dynamik die durchschnittliche Weglänge, die innerhalb einer Prozesslandschaft während der Simulation durchlaufen wird. Zeigt sich dabei ein eher niedriger Wert (beispielsweise im unteren Drittel einer gegebenen Skala), so ist dies ein Zeichen dafür, dass ein kurzer Pfad relativ häufig durchlaufen wird.

2.4 Vergleich mit existierenden Modellierungs- und Analyseansätzen

Ausgangspunkt für einen Vergleich des Process Landscaping mit anderen Modellierungs- und Analyseansätzen ist die Menge der in Kapitel 1 identifizierten fünf Problemstellungen (siehe Seite 5), für die das Process Landscaping Unterstützung anbietet. Es wird diskutiert, ob und inwiefern andere Methoden diese Problemstellungen ebenfalls berücksichtigen. Zusätzlich wird untersucht, welche verschiedenen Aspekte zu Vorgehen, Rollen, Ergebnissen und unterstützenden Techniken, die in der Literatur auch als Komponenten einer Methode bezeichnet werden [Hey95, HB96], im Vorder-

grund der jeweiligen Ansätze stehen. Dies ist insbesondere vor dem Hintergrund zu sehen, dass der Begriff der Methode gegenwärtig sehr unterschiedlich verwendet wird und somit ein Vergleich verschiedener Ansätze nicht immer ohne weiteres möglich ist. Für die nachfolgende Diskussion sind aus jeder dieser Methodenkategorien, die sich aus der Berücksichtigung der oben aufgeführten Aspekte ergeben, einzelne Repräsentanten ausgewählt worden.

Viele Prozessmodellierungsmethoden legen den Schwerpunkt auf eine zugrundeliegende Notation, die jedoch laut der Definition aus dem Informatik-Begriffsnetz [GF01] nur einen von mehreren Bestandteilen einer Methode ausmacht. Wenn Abeysinghe und Phalp beispielsweise die Kombination zweier Prozessmodellierungsmethoden diskutieren [AP97], sind dies keine Methoden wie im Rahmen dieser Arbeit formuliert (vgl. Einleitung dieses Kapitels 2), sondern lediglich Notationen (im genannten Beispiel Hoare's Communicating Sequential Processes (CSP) [Hoa85] und eine Untermenge der Role Activity Diagrams (RADs) [OR86]), die wiederum Bestandteil einer Methode sein können. Ziel der in [AP97] vorgestellten Methodenkombination ist es, zunächst mit RADs ein lesbares und verständliches Prozessmodell für Prozessbeteiligte zu erstellen, um die semantische Korrektheit der Modelle diskutieren zu können. Anschließend soll die Übersetzung nach CSP dazu dienen, ein Modell zur Prozesssimulation zu erhalten, um die Auswirkungen von Prozessveränderungen untersuchen zu können. Das Process Landscaping wird dagegen durchgängig von nur einer Notation, den Petrinetzen (vgl. Kapitel 3), unterstützt. Bei dieser wird für einen schnellen und transparenten Überblick zunächst von einigen Petrinetz-Eigenschaften abstrahiert, die in einer späteren Modellierungsphase für detaillierte Analysen wieder hinzugefügt werden.

Insgesamt berücksichtigt die Methode von Abeysinghe und Phalp keine der fünf in Abschnitt 1.1 aufgeführten Problemstellungen explizit. Schnittstellen zwischen Prozessmodellen werden hier lediglich über gleichnamige Aktivitäten identifiziert; Verteilungs- und Kommunikationsaspekte werden ebenso wie die Betrachtung von (Software-)Prozessen unter verschiedenen Blickwinkeln nicht angesprochen. Mit RADs kann man zwar auch unstrukturierte Prozesse abbilden, dies wird aber mehr durch die Notation unterstützt als durch die Vorgehensweise einer Methode.

Andere Methoden betreffen Softwareprozessverbesserungen (Software Process Improvement, SPI). Dies können Analysemethoden sein, die sich auf konkrete Eigenschaften von Prozessen beziehen und für diese Optimierungsansätze bieten, wie das beim Process Landscaping für die verschiedenen Kommunikationseigenschaften der Fall ist. Meist werden darunter jedoch spezielle Methoden wie SPICE [Sim96] und BOOTSTRAP [KSK+94] verstanden, oder es werden sogenannte *maturity models* wie CMM [PWCC94] zur Prozessverbesserung eingesetzt. Sie konzentrieren sich auf Prozesseigenschaften zur Unterstützung von Unternehmensstrategien und -organisationen und versuchen, für diese einen Zielerreichungsplan zu entwickeln [CFS01]. Beispiele aus diesem Bereich, die auf den oben genannten Methoden und Modellen aufsetzen, sind u.a. Accelerator, ein Modell und eine Methode zur Einführung eines Verbesserungsprogramms für Prozesse [IPB01] und eine Methode von Kellner et al. für Design, Definition und Entwicklung von Softwareprozessen [KBO96].

Der erste Schritt der Accelerator-Methode ist die Evaluation von Unternehmensprozessen auf der Basis von Dokumenten-Reviews, Interviews und Informationsanalyse. Ob und wie das Evaluationsergebnis als eine Menge von Prozessmodellen dokumentiert wird, bleibt dabei den Anwendern von Accelerator überlassen; eine (durchgängige) Modellierungsunterstützung wird nicht angeboten. Während der Evaluationsphase stehen nicht die Schnittstellen zwischen Prozessen bzw. Aktivitäten im Vordergrund der Betrachtung, sondern die Koordination parallel durchzuführender Aktivitäten. Auch die geografische Verteilung der Prozessbeteiligten und die daraus resultierende Auswirkung auf deren Kommunikationseffizienz wird nicht explizit untersucht.

Kellner, Briand und Over sehen die Erstellung von Prozessmodellen als wichtige Voraussetzung zur Erreichung von Prozessverbesserungen an [KBO96]. Schwerpunkte ihrer Methode liegen daher auf einer geeigneten Vorgehensweise zum Design und zur Definition von Softwareprozessen und auf der Institutionalisierung einer kontinuierlichen Prozessverbesserung. Aber auch hier wird keine explizite Modellierungsunterstützung angeboten. Stattdessen verweisen die Autoren unter anderem auf [CKO92], wo bewusst auf die Empfehlung eines Modellierungsparadigmas verzichtet wird, da ihrer Meinung nach jedes Paradigma nur die Darstellung bestimmter Aspekte von Softwareprozessen unterstützt. Aus diesem Grund werden auch bei ihrer Methode weder Schnittstellen noch Verteilungsaspekte explizit betrachtet, und es fehlt eine gezielte Unterstützung für unstrukturierte Prozesse. Insgesamt betrachten damit beide Methoden, die den Fokus auf Softwareprozessverbesserungen setzen, keine der Problemstellungen, die das Process Landscaping aufgreift. Lediglich das Ziel der Prozessverbesserung bezüglich bestimmter Eigenschaften kann als Gemeinsamkeit festgehalten werden.

Bayer, Junginger und Kühn schlagen (für die Entwicklung von e-business-Anwendungen) eine Modellierungsmöglichkeit vor, die Schnittstellen zu Informationssystemen zwar berücksichtigt, aber den Schwerpunkt eher auf die Schnittstellen zwischen Kunde und e-business-Anwendung sowie die im Rahmen eines Entwicklungsprozesses zu erstellenden Teilprodukte setzt [BJK00]. Ihr Ansatz ist stark werkzeugzentriert, da sie mit dem „(meta) business process management toolkit ADONIS“ den Modellierer unterstützen wollen, ohne detailliert auf eine dem Werkzeug und seiner Handhabung zugrundeliegende Methode einzugehen. Bei dieser Methode zur „Metamodellierung im Geschäftsprozeßmanagement“ wird die Menge der methodischen Schritte zur Erstellung eines Prozessmodells als Vorgehensmodell bezeichnet und die Notation als eine – von der Methode selbst unabhängige – Modellierungstechnik [KJKP99]. Da für die Modellierung sowohl ereignisgesteuerte Prozessketten [LSW97] als auch Aktivitätsdiagramme [Bal01] eingesetzt werden können, unterstützt die Methode die Darstellung strukturierter und unstrukturierter Prozesse. Eine durchgängige Modellierungsunterstützung, die Inkonsistenzen beim Notationswechsel vermeidet, ist jedoch nicht vorhanden. Dies verhindert eine Modellerstellung nach der Process Landscaping Methode, die im Rahmen eines Modellierungsprojektes Ablaufinformationen erst zu einem späteren Zeitpunkt berücksichtigen will.

Die explizite Berücksichtigung der geografischen Verteilung eines Prozesses ist bei der

von Junginger et al. erarbeiteten Methode ebenfalls nicht gegeben; eine hierfür gesonderte Sicht auf Prozessmodelle wird nicht zur Verfügung gestellt. Allgemein sehen die Autoren die Erstellung verschiedener Sichten jedoch als wichtige Forschungsaufgabe im Rahmen der Weiterentwicklung ihrer Methode an [KJKP99]. Die bei den genannten Autoren diskutierten Anforderungen an eine Analyse sind auf einem anderen Abstraktionsniveau als beim Process Landscaping zu sehen. Sie bieten im Rahmen ihrer Methode die Möglichkeit sowohl der statischen als auch der dynamischen Analyse an. Beide Analyseformen beziehen sich jedoch nicht auf konkrete Ziele, wie dies durch die schwerpunktmäßige Betrachtung von Kommunikationsstrukturen und -verhalten beim Process Landscaping der Fall ist.

Zwei weitere Methoden, die sich nicht auf eine Werkzeugunterstützung oder einzelne Methodenbestandteile konzentrieren, werden nachfolgend ebenfalls mit dem Process Landscaping verglichen. Die PTA (Process Tradeoff Analysis) Methode ist ein quantitativer Modellierungsansatz zur Unterstützung von Planungs- und Kontrollaktivitäten des Projektmanagements, die potentielle Prozessveränderungen wie Entwicklungskosten, Produktqualität und Projektplanung evaluieren [RHV00]. Auf der Grundlage stochastischer, zustandsbasierter Simulationsmodelle werden verschiedene mögliche Prozessalternativen gegeneinander abgewogen. Ein ähnliches Vorgehen ist beim Process Landscaping für potentielle Veränderungen einer Kommunikationsinfrastruktur gewählt worden. Die Identifikation der bestmöglichen Prozessalternative im Hinblick auf zukünftige Prozessveränderungen wird beim Process Landscaping jedoch auch für die oberen Ebenen einer Prozesslandschaft unterstützt, für die mit PTA aufgrund der fehlenden Dynamik keine Simulationsmodelle erstellt und ausgewertet werden können. Für detailliert modellierte, strukturierte Prozesse liegt der Analyseschwerpunkt von PTA auf der Auswertung von Daten bezüglich Kosten, Zeiten und Qualität für ein Softwareentwicklungsprojekt bzw. ein zu erstellendes Softwareprodukt [RVM99]. Raffo et. al berücksichtigen zwar explizit die Abhängigkeiten zwischen verschiedenen Prozesskomponenten, dies erfolgt jedoch nicht über die Betrachtung von Schnittstellen, sondern über ein gemeinsames Datenrepository. Kommunikationsaspekte werden nicht explizit betrachtet, geografische Verteilungsaspekte eines Softwareprozesses ebenfalls nicht.

SeeMe präsentiert sich als Modellierungsmethode „für die Darstellung sozialer und semi-strukturierter Aspekte von Kommunikations- und Kooperationsbeziehungen“. Dabei liegt ein Schwerpunkt auf der „Vermeidung hinderlicher Einschränkungen der Modellierungsfreiheit sowie der Darstellung vager Informationen“ [HHL98]. Diese Methode ist jedoch nicht für den Anwendungsbereich der Geschäftsprozessmodellierung oder – noch spezieller – für den der Softwareprozessmodellierung entwickelt, sondern für den Bereich sozialer Prozesse mit Schwerpunkt auf Kommunikation. Aufgrund dieser völlig unterschiedlichen Einsatzgebiete erfährt auch der Begriff der Kommunikation eine gänzlich andere Betrachtung. Während SeeMe Prozessmodelle als Kommunikationsmittel u.a. zur Aushandlung von Softwareanforderungen ansieht und daher ein Optimum ihrer Kommunizierbarkeit anstrebt, nutzt das Process Landscaping eine Prozesslandschaft zur Analyse von Kommunikationsstruktur und -verhalten inner-

halb der dargestellten Prozesse. Daher sind das Process Landscaping und SeeMe bzgl. ihrer Modellierungsziele nur schwer zu vergleichen. Lediglich die Anforderung, Informationen bei der Modellierung bewusst auszulassen, kann als gemeinsames Merkmal eingestuft werden: Während das Process Landscaping auf den oberen Ebenen einer Prozesslandschaft bewusst auf die Darstellung von Ablaufinformationen verzichtet, hebt SeeMe diese Einschränkungen der Modellierungsfreiheit durch die Bereitstellung zusätzlicher Notationselemente auf, die dem Modellierer das bewusste Auslassen von Informationen sowie die Kennzeichnung unsicherer Informationen ermöglichen [HKL02]. Unstrukturierte Prozesse können also mit beiden Methoden abgebildet werden, somit bieten beide einen Lösungsansatz zur vierten in Abschnitt 1.1 diskutierten Problemstellung an. Das beim Process Landscaping als elementar angesehene Prozesselement der Schnittstelle wird bei der Modellierung mit SeeMe jedoch nicht eingesetzt. Des Weiteren werden die Problemstellungen bezüglich der geografischen Verteilung der Prozessbeteiligten und der Erstellung verschiedener Sichten auf eine Prozesslandschaft mit SeeMe nicht berücksichtigt.

Hess und Brecht untersuchen insgesamt 17 Methoden „für die projekthafte, grundlegende Neugestaltung betrieblicher Prozesse“ aus unterschiedlichen „Denkrichtungen“ und Umfeldern [HB96]. Sie stellen dabei ebenfalls gravierende Unterschiede sowohl in den Schwerpunkten dieser Methoden (Ablauforganisation, Anwendungsdomäne, unterstützende Informationssysteme) als auch in ihrem Aufbau (Vorgehen in Reorganisationsprojekten, Ausarbeitung von Modellierungstechniken) fest. Ihr Fazit: „Eine einheitliche Konstruktionslehre für Prozesse hat sich noch nicht herausgebildet“.

Für den Vergleich der Methoden haben die Autoren einen Kriterienkatalog erstellt, der verschiedene Gesichtspunkte von Gestaltungsfeldern über die Einbeziehung der Rolle der Geschäftsstrategie bis zur Gegenüberstellung der Methodenkomponenten enthält. Die Kriterien, auf deren Grundlage das Process Landscaping entstanden ist, spielen für den Vergleich von Hess und Brecht jedoch keine Rolle. Umgekehrt enthält ihr Katalog aber Kriterien, an denen sich auch das Process Landscaping messen lassen kann:

- Die bei den Autoren als *Umfang der Prozessarchitektur* bezeichnete Menge der betrachteten Prozesse beinhaltet beim Process Landscaping alle Prozesse eines Unternehmens.
- Die Ist-Dokumentation der betrachteten Prozesse kann beim Process Landscaping flexibel von *grob*, über *in Teilbereichen detailliert* bis *detailliert* gestaltet werden.
- Das generelle Vorgehen der Methode ist als Top-Down einzustufen.
- Die Modellierung des Ablaufs beinhaltet beim Process Landscaping sowohl zeitliche Abhängigkeiten als auch den Datenfluss zwischen Aktivitäten.
- Als Methodenkomponenten des Process Landscaping sind das strukturierte Vorgehen, beteiligte Rollen, verschiedene Techniken zur Wissensakquise, Modellierung und Analyse sowie die explizit zu erreichenden Ziele bzw. Ergebnisse vorhanden.

Der Vergleich des Process Landscaping mit anderen Modellierungs- und Analysemethoden macht deutlich, dass der Begriff der Methode auch aktuell noch sehr unterschiedlich verwendet wird. Beim Entwurf einer Methode wird zumeist nur eine Teilmenge ihrer Bestandteile (Notation, systematische Handlungsanweisungen und Regeln zur Überprüfung der Ergebnisse), ihr Ziel oder aber eine Werkzeugunterstützung schwerpunktmäßig betrachtet. Das Process Landscaping verfolgt dagegen einen Ansatz, der alle aufgeführten Bestandteile berücksichtigt. Tabelle 2.1 fasst die vergleichende Diskussion zusammen.

Da nicht alle diskutierten Methoden einen Namen haben, ist in der ersten Spalte der Tabelle zunächst eine Referenz angegeben, um eindeutig auf die gerade Betrachtete verweisen zu können. Die Methodennamen sind zusätzlich angegeben, soweit sie existieren. Die zweite Spalte gibt jeweils den individuellen Schwerpunkt einer Methode an, und die dritte Spalte enthält die Nummer der in Abschnitt 1.1, Seite 5 aufgeführten Problemstellungen, die neben dem Process Landscaping auch von der jeweils diskutierten Methode berücksichtigt werden. Die letzte Spalte enthält zusätzliche Gemeinsamkeiten oder Unterschiede, die durch einen ausschließlichen Vergleich entlang der fünf Problemstellungen nicht deutlich werden.

Die für die angesprochene Betrachtung des Verteilungsaspektes erforderliche explizite Berücksichtigung von Schnittstellen von Beginn der Modellierung an ist ein Alleinstellungsmerkmal des Process Landscaping. Dieses unterstützt unter anderem die Erstellung der logischen Sicht auf eine Prozesslandschaft, in der direkte Schnittstellen zu einem Kunden und Schnittstellen von Prozessen entlang des Kundenlebenszyklus betrachtet werden. Für letzteres ist hervorzuheben, dass die Schnittstellen zwischen Prozessen hier ohne die Berücksichtigung organisatorischer Strukturen eines Unternehmens betrachtet werden. Diese rein logische Sicht erlaubt somit die Identifikation von Optimierungspotential, ohne dabei von vorhandenen Organisationsstrukturen eingeschränkt zu werden.

Durch die Art der Berücksichtigung des Verteilungsaspektes in einer Softwareprozesslandschaft werden auch die Kommunikationsinfrastruktur und das Kommunikationsverhalten in einer solchen Landschaft auf eine Weise betrachtet, wie dies in der Literatur bislang nicht bekannt ist. Dies wird jedoch nicht erst durch den Vergleich mit SeeMe deutlich, sondern bereits an der Verwendung des Kommunikationsbegriffs, der sich stark von der üblichen Verwendung im Bereich der Informatik abhebt (vgl. Abschnitt 1.3).

Auch die Forderung nach verschiedenen Blickwinkeln auf eine Prozesslandschaft ohne den Aufwand einer Neumodellierung oder einer Integration verschiedener Sichten ist bislang in der Literatur nicht diskutiert worden. Die Notwendigkeit der Existenz verschiedener Sichten ist zwar allgemein anerkannt, letztere wurden jedoch – zumindest aufgrund des Kenntnisstandes durch vorangegangene Literaturrecherche – bislang immer mit dem erwähnten Zusatzaufwand erstellt. Zwar lässt sich darüber streiten, wie groß bzw. wie gering der Aufwand einer Umstrukturierung ist, die bei anderen Ansätzen bestehende Gefahr von Inkonsistenzen ist jedoch beim Process Landscaping nicht gegeben, da keine der in einer logischen Sicht vorhandenen

Methoden-referenz	**Schwerpunkt**	**berücksichtigte Problemstellung**	**Gemeinsamkeiten / Unterschiede**
[AP97]	Notation	4.	Darstellung unstrukturierter Prozesse aufgrund der Notation möglich; keine durchgängige Notation; keine methodische Vorgehensweise
[IPB01] Accelerator	SPI	–	gemeinsames Ziel der Prozess-verbesserung
[KBO96]	SPI	–	Modellierung von Prozessen als gemeinsame Voraussetzung ihrer Verbesserung
[KJKP99] Adonis	Werkzeugunterstützung	4.	Schnittstellenbetrachtung implizit, aber zwischen Informationssystemen und Kunden
[RHV00] PTA	Projektmanagement	–	ähnliches Vorgehen bei der Prozess-analyse durch Simulation und Gegen-überstellung verschiedener Prozess-alternativen
[HHL98] SeeMe	Kommunikationsaspekte	4., 5.	Betrachtung unterschiedlicher Kommunikationsaspekte
[HB96] 17 Methoden	betriebliche Prozesse	–	Beschreibung der Vorgehensweise als gemeinsame Methodenkomponente

Tabelle 2.1: Vergleich des Process Landscaping mit anderen Methoden

Aktivitäten und Schnittstellen in der lokalen Sicht verloren geht. Sie werden lediglich umgestellt und bezüglich ihrer geografischen Verteilung detaillierter dargestellt. Eine formale Basis bietet hierzu definierte Regeln zur Umstrukturierung und ermöglicht Konsistenzprüfungen (vgl. Abschnitt 3.2.2.1).

Ein weiterer wesentlicher Unterschied zwischen Process Landscaping und anderen Modellierungsmethoden liegt darin, dass bislang vorrangig Aspekte wie Kontrollfluss, Datenfluss und Arbeitsverteilung im Hinblick auf den Ablauf von Prozessen abgebildet wurden. Das Process Landscaping stellt dagegen den bislang in der Literatur vernachlässigten Verteilungsaspekt von Prozessen und deren Kommunikation untereinander in den Vordergrund der Betrachtung, und zwar unabhängig davon, ob dabei ein Ablauf berücksichtigt wird oder nicht [GW01b, GW01a]. Die bislang gemachte Differenzierung zwischen wiederholbaren, strukturierten Prozessen und den eher unstrukturierten, kreativen und kooperativen Prozessen [CFJ98] ist mit dem Process Landscaping abgeschwächt worden, da die Methode die Modellierung beider Prozessformen ermöglicht und außerdem einen konsistenten Übergang von den Prozessmodellen der unstrukturierten zu denen der strukturierten Prozesse bietet. Dazu wird eine Prozesslandschaft nach dem Top-Down-Vorgehen erstellt, wobei auch strukturierte Prozesse zunächst relativ abstrakt modelliert werden und diese dann sukzessive verfeinert werden können. Diese Vorgehensweise wird durch eine formale Notation unterstützt; die Sicherung des konsistenten Übergangs auf Basis dieser formalen Notation ist in Abschnitt 3.2.1 dargestellt.

In diesem Kapitel sind die Notation und eine Werkzeugunterstützung jedoch noch nicht diskutiert worden. Die Notwendigkeit der formalen Basis ist in Kapitel 1 bereits motiviert worden. Sie wird im nachfolgenden Kapitel 3 detailliert diskutiert.

Kapitel 3

Formale Basis des Process Landscaping

Die Möglichkeit der formalen Beschreibung einer verteilten Prozesslandschaft ist für das Process Landscaping eine unverzichtbare Voraussetzung, insbesondere zur Durchführung der in Abschnitt 2.3 diskutierten Analyseschritte der Methode. Erst eine formale Notation erlaubt exakte Konsistenzprüfungen einer erstellten Prozesslandschaft sowie genaue Analysen von Landschaftseigenschaften. Im Rahmen des Process Landscaping ergeben sich jedoch noch weitere Anforderungen an eine Beschreibungssprache. Dies sind:

- die Möglichkeit der expliziten Modellierung von Schnittstellen,
- die Unterstützung eines unterschiedlichen Detaillierungsgrades von Landschaftselementen,
- das Abstrahieren von Ablaufinformationen für die oberen Ebenen einer Prozesslandschaft,
- die Berücksichtigung des Verteilungsaspekts innerhalb einer Prozesslandschaft und
- die Formulierung von Attributen der Landschaftselemente zur Analyse verschiedener Landschaftseigenschaften.

Insbesondere die vier zuerst genannten Anforderungen gelten auch für eine grafische Darstellung einer Prozesslandschaft. Alle möglichen Attribute von Landschaftselementen visualisieren zu wollen, würde jedoch die Komplexität der grafischen Darstellung zu groß werden lassen.

Eine Grundidee des Process Landscaping ist es, die Modellierung einer Prozesslandschaft von einem Abstraktionsniveau aus zu beginnen, das zunächst von den Ablaufinformationen von und zwischen Prozessen abstrahiert und sich auf die Identifikation

von (Kern-)Aktivitäten und zentralen Informationstypen konzentriert. Eine solche Prozesslandschaft muss später möglichst einfach um Ablaufinformationen und weitere Informationstypen ergänzt werden können. Gleichzeitig soll eine Übersicht über die Hierarchiebeziehungen zwischen den verschiedenen Abstraktionsebenen dem Modellierer helfen, den Gesamtüberblick über die entstehende Prozesslandschaft zu behalten. Für die Unterstützung der Process Landscaping Methode ist eine formale Grundlage erforderlich, die für sehr unterschiedliche Abstraktionsebenen geeignet oder geeignet erweiterbar ist.

Die Auswahl für eine geeignete Prozessmodellierungssprache zur Beschreibung einer Prozesslandschaft ist mit den aufgeführten Anforderungen bereits stark eingeschränkt: Semi-formale Sprachen wie UML [FR99] können zwar für die Erstellung, nicht aber für die Analyse einer Prozesslandschaft eingesetzt werden. Streng formale Sprachen wie Z [Spi92] sind dagegen nicht geeignet, Prozesse so abzubilden, dass die resultierenden Modelle auch für Prozessbeteiligte ohne genaue Kenntnisse dieser Spezifikationssprache leicht verständlich sind. Sprachen wie JIL [SO97] oder APPL/A [SHO95] unterstützen keine Visualisierung von Prozessen, sondern setzen ihren Schwerpunkt auf die Unterstützung der Prozessausführung. Einen Überblick über existierende Prozessmodellierungssprachen geben Zamli und Lee [ZL01].

In diesem Kapitel wird gezeigt, dass Petrinetze [Bau96] für die formulierten Anforderungen zweckmäßig sind. Sie sind ein gut geeignetes Beschreibungsmittel für verteilte (Software-)Prozesse und vereinen die Vorteile eines grafischen Beschreibungsmittels und eines mathematisch präzisen und daher analysierbaren Formalismus.

Die Modellierung einer Prozesslandschaft startet zunächst mit einer – im Rahmen dieser Dissertation entwickelten – abstrakten Petrinetz-Variante PLL (Process Landscaping Language), mit der die oberen Ebenen einer Prozesslandschaft relativ abstrakt und ohne Ablaufinformationen, aber bereits mit Hierarchieinformationen und ersten Relationen zwischen verschiedenen Landschaftselementen beschrieben werden können. Diese abstrahierten Ablaufinformationen können über mehrere Schritte nachträglich ergänzt werden, wenn beispielsweise das dynamische Verhalten der Prozesslandschaft analysiert werden soll.

Zur Abgrenzung der einzelnen, aufeinander aufbauenden Erweiterungsschritte untereinander werden verschiedene Ausbaustufen einer Prozesslandschaft eingeführt. Die Basisdefinition einer Prozesslandschaft ermöglicht die Darstellung ihrer oberen Ebenen, die erste Ausbaustufe enthält zusätzliche Informationen beispielsweise über die lokale Verteilung ihrer Elemente. Die zweite Ausbaustufe ermöglicht die Analyse von Eigenschaften einer Prozesslandschaft, die ihre Kommunikationsinfrastruktur charakterisieren. In der dritten Ausbaustufe einer Prozesslandschaft ist diese um die Möglichkeit der Modellierung von Ablaufinformationen erweitert, und in der vierten Ausbaustufe können Analysen bezüglich des Kommunikationsverhaltens einer Prozesslandschaft durchgeführt werden.

Die Struktur dieses Kapitels folgt dem Top-Down-Vorgehen der Modellierung einer Prozesslandschaft mit Process Landscaping. Zunächst werden die zur Modellierung (Abschnitt 3.1.1) und grafischen Repräsentation (Abschnitt 3.1.2) der oberen Land-

schaftsebenen erforderlichen Strukturen und Konzepte definiert. Diese werden anschließend um notwendige Definitionen zur Analyse von Kommunikationseigenschaften erweitert (Abschnitt 3.1.3). Abschnitt 3.2.1 diskutiert die verschiedenen Schritte zur nachträglichen Ergänzung von Ablaufinformationen in eine Prozesslandschaft. In Abschnitt 3.2.2 werden die in Abschnitt 3.1.1 eingeführten Strukturen und Konzepte für die Modellierung der unteren Landschaftsebenen erweitert. Sie bilden die zur Ausführung und damit zur dynamischen Analyse einer Prozesslandschaft notwendigen formalen Grundlagen. Die Definitionen werden im gesamten Kapitel – soweit sinnvoll – von dem in Abschnitt 2.2.1 eingeführten Beispiel durchgängig begleitet, anhand dessen der Einsatz und Nutzen der verschiedenen Sprachkonstrukte verdeutlicht wird. Abschließend wird in Abschnitt 3.3 eine vergleichende Diskussion mit anderen aus der Literatur bekannten Petrinetz-Ansätzen geführt.

3.1 Notation der oberen Ebenen einer Prozesslandschaft

Im Folgenden wird die Petrinetz-Variante PLL eingeführt, die eine Modellierung von Prozessen (als komplexe Aktivitäten) und Dokumenten (als zentrale Informationstypen) sowie ihren Beziehungen zueinander erlaubt, jedoch explizit von Ablaufinformationen abstrahiert. Es wird gezeigt, dass sie die Anforderungen an die Notation von oberen Ebenen einer Prozesslandschaft erfüllt. Abschnitt 3.1.1 definiert zunächst den Kern von PLL, mit dem das Konzept der Schnittstellenbetrachtung sowie die Verfeinerungsmöglichkeiten von Schnittstellen und Aktivitäten innerhalb einer Prozesslandschaft unterstützt werden. Die erste Ausbaustufe einer Prozesslandschaft erlaubt unter anderem die Berücksichtigung von Verteilungsaspekten und unterscheidet verschiedene Dokumenttypen. Verschiedene grafische Repräsentationen einer Prozesslandschaft werden in Abschnitt 3.1.2 eingeführt. Dabei wird insbesondere auf Unterschiede zur grafischen Repräsentation konventioneller Petrinetze eingegangen. In Abschnitt 3.1.3 wird das der ersten Ausbaustufe einer Prozesslandschaft zugrundeliegende Petrinetz um eine Menge von Abbildungen und Begriffen erweitert, die die Analyse der oberen Landschaftsebenen bezüglich verschiedener Kommunikationseigenschaften ermöglicht (zweite Ausbaustufe einer Prozesslandschaft).

3.1.1 Basisdefinition und erste Ausbaustufe einer Prozesslandschaft

Für die Einführung von PLL ist die Kenntnis über einige Relationen und Mengen erforderlich, die nachfolgend definiert werden. Mit Hilfe der Menge von Nachfolgern und der Menge von Vorgängern eines Elementes x aus einer Menge X können Aussagen über die in der Einleitung erwähnten hierarchischen Beziehungen zwischen Aktivitäten auf den verschiedenen Ebenen einer Prozesslandschaft formuliert werden.

Definition 3.1.1. *Sei $R \subseteq X \times X$ eine binäre Relation über der Menge X.*

$$R(\{x\}) := \{y \in X \mid (x, y) \in R\}$$

ist das Relationenbild der Relation R für $x \in X$. Dabei stellt $R(x)$ eine vereinfachende Schreibweise für $R(\{x\})$ dar.

Das Relationenbild der Relation R beschreibt also diejenigen Elemente $y \in X$, die mit $x \in X$ in Relation R stehen.

Definition 3.1.2. *Sei X eine endliche nichtleere Menge und R eine Menge geordneter Paare (x, y) mit $x, y \in X$, also $R \subset X \times X$. Dann heißt $G = (X, R)$ ein **endlicher gerichteter Graph**. Die Elemente von X heißen **Knoten**, die Elemente von R heißen **Kanten**. Ist (x, y) eine Kante, so heißt x **Vorgänger** von y, und y heißt **Nachfolger** von x. Ein Knoten, der keinen Vorgänger hat, heißt **Quelle**; ein Knoten, der keinen Nachfolger hat, heißt **Senke** oder **Blatt**.*

Definition 3.1.3. *Ein gerichteter Graph mit genau einer Quelle und der Eigenschaft, dass es für jeden Knoten $x \in X$ genau einen Weg von der Quelle nach x gibt, heißt **gerichteter Baum**.*

Definition 3.1.4. *Sei G ein gerichteter Baum mit $G = (X, R)$. Die **Menge der Nachfolger** $succ_R(x)$ eines Elements x aus der Menge X ist wie folgt definiert:*

$$succ_R(x) := R(x)$$

$succ_R(x)$ bezeichnet die Menge aller direkten Nachfolger von x.

Bemerkung 3.1.5. *$succ_R^+(x)$ ist die transitive Hülle von $succ_R(x)$ und bezeichnet die Menge aller Nachfolger – auch der nicht direkten – von x.*

Definition 3.1.6. *Sei G ein gerichteter Baum mit $G = (X, R)$. Die **Menge der Vorgänger** $pred_R(x)$ eines Elements x aus der Menge X ist wie folgt definiert:*

$$pred_R(x) := \{y \in X \mid x \in succ_R(y)\}$$

Die Vorgänger eines Elementes x können also über eine Menge von Nachfolgern definiert werden: Ist das Element x in der Menge der Nachfolger des Elementes y, dann ist y Vorgänger von x.

Notation 3.1.7. *Wenn keine Missverständnisse auftreten können, wird im Folgenden $succ_R$ kurz als $succ$ und $pred_R$ kurz als $pred$ geschrieben. Entsprechend wird $succ_R^+$ kurz als $succ^+$ geschrieben. Des Weiteren wird ein gerichteter Baum kurz als Baum bezeichnet.*

Nachfolgend werden ausgezeichnete Mengen innerhalb des Relationenbildes definiert.

Definition 3.1.8. *Sei G ein Baum mit $G = (X, R)$. Die Abbildung **ref** (kurz für refinement) liefert alle Elemente der Menge X, deren Nachfolgermenge nicht leer ist:*

$$ref(R) := \{x \in X \mid succ(x) \neq \emptyset\}$$

*Die Abbildung **leaves** liefert alle Elemente der Menge X, deren Nachfolgermenge leer ist, also alle Blätter des Baumes G:*

$$leaves(R) := \{x \in X \mid succ(x) = \emptyset\}$$

*Die Abbildung **top** liefert alle Elemente der Menge X, die direkte Nachfolger der Quelle $r \in X$ des Baumes G sind:*

$$top(R) := succ(r)$$

Mit dem Baum $G = (X, R)$ ist es möglich, hierarchische Strukturen innerhalb einer Prozesslandschaft zu beschreiben und über seine grafische Repräsentation einem Modellierer auf einfache Art und Weise einen Gesamtüberblick über diese Landschaft zu ermöglichen. Mit den definierten Teilmengen der Menge X können konkrete Abstraktionsebenen einer Prozesslandschaft spezifiziert werden.

Definition 3.1.9. *Ein **Petrinetz** (oder Petrinetzgraph) ist ein Tripel PN = (S,T,F) mit*

$$S \cap T = \emptyset$$
$$F \subseteq (S \times T) \cup (T \times S)$$

*Die Elemente von S heißen **Stellen**, die von T heißen **Transitionen**. Die Elemente beider Mengen, $S \cup T$, werden auch als **Knoten** bezeichnet. Die Elemente von F heißen **Kanten**.*

Bemerkung 3.1.10. *Ein Petrinetz ist hier lediglich ein bipartiter Graph, da seine Knotenmengen disjunkt sind und jede Kante einen Knoten aus S mit einem Knoten aus T verbindet.*

Definition 3.1.11. *Die **Basis einer Prozesslandschaft** ist definiert als ein Tupel*

$$\omega := (A, S, Z, AB) \qquad \textit{mit}$$

$$A \neq \emptyset, S \neq \emptyset$$
$$A \cap S = \emptyset$$
$$Z \subseteq (A \times S) \cup (S \times A)$$

Der Graph (A, AB) ist ein Baum mit $AB \subset A \times A$.
Das Tripel $\Big(leaves(AB), S, Z \cap \big((leaves(AB) \times S) \cup (S \times leaves(AB))\big)\Big)$ ist ein Petrinetz.

Notation 3.1.12. *Elemente der Menge A heißen **Aktivitätsbeschreibungen (kurz: Aktivitäten)** und Elemente der Menge S **Schnittstellenbeschreibungen (kurz: Schnittstellen)**. Elemente der Menge Z heißen **Zugriffe**. $(a, s) \in Z$ bedeutet, dass eine Aktivität $a \in A$ einen erzeugenden oder schreibenden Zugriff auf eine Schnittstelle $s \in S$*

*hat, und $(s, a) \in Z$ bedeutet, dass eine Aktivität $a \in A$ lesend auf eine Schnittstelle $s \in S$ zugreift. Die Relation AB ist eine ausgezeichnete Menge von Paaren von Aktivitäten. Sie ermöglicht eine hierarchische Ordnung der Aktivitäten als Baum. Der Baum (A, AB), im Folgenden verkürzt als AB bezeichnet, heißt daher **Aktivitätenbaum**. So kann über den Aktivitätenbaum zum Ausdruck gebracht werden, dass eine Aktivität aus mehreren Nachfolgeraktivitäten zusammengesetzt ist.*

Notation 3.1.13. *Die Menge $leaves(AB)$, deren Elemente als Aktivitäten einer Prozesslandschaft im zugehörigen Petrinetz abgebildet werden, wird nachfolgend verkürzt als $\mathbf{A_{PN}}$ bezeichnet.*
Analog wird die Menge $Z \cap \big((leaves(AB) \times S) \cup (S \times leaves(AB))\big)$, deren Elemente die Zugriffe einer Prozesslandschaft im zugehörigen Petrinetz darstellen, nachfolgend verkürzt als $\mathbf{Z_{PN}}$ bezeichnet.

Wichtig ist festzuhalten, dass eine Prozesslandschaft nur in der Kombination eines Petrinetzes mit einem Aktivitätenbaum vollständig beschrieben werden kann, deren Aktivitäten aus einer gemeinsamen Menge A stammen: Während der Aktivitätenbaum die hierarchische Struktur einer Prozesslandschaft beschreibt und durch die Relationen zwischen den verschiedenen Aktivitäten auch die Anzahl der Abstraktionsebenen bestimmt, beschreibt ein Petrinetz die Struktur genau einer Abstraktionsebene, die – zumeist – ausschließlich die Blätter des Aktivitätenbaumes zusammen ihren Zugriffen auf alle innerhalb der Landschaft existierenden Schnittstellen darstellt.

Die Verfeinerung von Aktivitäten einer Prozesslandschaft ist ein elementarer Modellierungsschritt des Process Landscaping. Die zur Darstellung der Prozesslandschaft verwendete Notation muss daher die Möglichkeit der Verfeinerung entsprechend berücksichtigen. Nachfolgend wird definiert, wie sich die Verfeinerung einer Aktivität sowohl auf das zugrundeliegende Petrinetz als auch auf den zugehörigen Aktivitätenbaum auswirkt.

Definition 3.1.14. *Sei $\omega = (A, S, Z, AB)$ die Basis einer Prozesslandschaft und $a \in leaves(AB)$. Eine Prozesslandschaft*

$$\omega' = (A', S', Z', AB') \text{ mit } \quad A' = A \cup \{a_1, ..., a_n\} \text{ und } A \cap \{a_1, ..., a_n\} = \emptyset, n \geq 2$$

*heißt **Verfeinerung** von ω **bezüglich der Aktivität** $\mathbf{a} :\Leftrightarrow$*

1. *$S' = S \cup \{s_1, ..., s_m\}$ und $S \cap \{s_1, ..., s_m\} = \emptyset, m \in \mathbb{N}_0$, wobei $m = 0$ bedeutet, dass $S' = S$*

2. *(a) Es existieren Mengen D_1, D_2 mit*

$$D_1 = \{(s, \tilde{a}) \mid \tilde{a} \in A \backslash \{a\}, s \in S \wedge (s, \tilde{a}) \in Z\}$$
$$D_2 = \{(\tilde{a}, s) \mid \tilde{a} \in A \backslash \{a\}, s \in S \wedge (\tilde{a}, s) \in Z\}$$

(b) Seien $\{s \in S \mid (s,a) \in Z\} = \{s_{in,1}, ..., s_{in,p}\}$ *für ein* $p \in \mathbb{N}_0$*. Dann existiert für* $p \neq 0$ *eine Menge* D_3 *mit*

$$D_3 = \bigcup_{1 \leq k \leq p} B_k \text{ wobei}$$
$$\forall\, 1 \leq k \leq p : B_k \subseteq \{(s_{in,k}, a_i) \mid 1 \leq i \leq n\} \text{ und } B_k \neq \emptyset$$

$p = 0$ *bedeutet, dass* $D_3 = \emptyset$.

(c) Seien $\{s \in S \mid (a,s) \in Z\} = \{s_{out,1}, ..., s_{out,q}\}$ *für ein* $q \in \mathbb{N}_0$*. Dann existiert für* $q \neq 0$ *eine Menge* D_4 *mit*

$$D_4 = \bigcup_{1 \leq k \leq p} C_k \text{ wobei}$$
$$\forall\, 1 \leq k \leq p : C_k \subseteq \{(a_i, s_{out,k}) \mid 1 \leq i \leq n\} \text{ und } C_k \neq \emptyset$$

$q = 0$ *bedeutet, dass* $D_4 = \emptyset$.

(d) Schließlich existieren Mengen D_5, D_6 *mit*

$$D_5 = \bigcup_{1 \leq j \leq m} F_j \text{ wobei}$$
$$\forall\, 1 \leq j \leq m : F_j \subseteq \{(s_j, a_i) \mid 1 \leq i \leq n\}$$
$$D_6 = \bigcup_{1 \leq j \leq m} G_j \text{ wobei}$$
$$\forall\, 1 \leq j \leq m : G_j \subseteq \{(a_i, s_j) \mid 1 \leq i \leq n\}$$
$$\text{und } \forall\, 1 \leq j \leq m : F_j \cup G_j \neq \emptyset$$

Es gilt: $Z' = \bigcup_{1 \leq l \leq 6} D_l$

3. $AB' = AB \cup \{(a, a_i) \mid 1 \leq i \leq n\}$

Bedingung 1. erlaubt das Hinzufügen weiterer Schnittstellen in die verfeinerte Prozesslandschaft. Bedingung 2. legt die Menge der Zugriffe Z' als Vereinigung verschiedener Teilmengen fest:

- Alle Zugriffe, die bereits in Z existieren, aber nicht mit der Aktivität a verbunden waren, bleiben erhalten (D_1, D_2).
- Es kommen keine Zugriffe auf die Aktivität a mehr in ω' vor. Diejenigen Zugriffe, die die Schnittstellen $s_{in,k}$ ($s_{out,k}$) mit der Aktivität a verbunden haben, werden in ω' durch solche ersetzt, die die $s_{in,k}$ ($s_{out,k}$) mit den verfeinernden Aktivitäten a_i verbinden (D_3, D_4). Sind D_3 bzw. D_4 nicht leer, wird jede Schnittstelle $s_{in,k}$ bzw. $s_{out,k}$ mit mindestens einer Aktivität a_i verbunden. An dieser Stelle werden also Elemente $z \in Z$ ausgetauscht. Welche der nichtleeren Teilmengen B_k und C_k verwendet werden, aus denen sich die Mengen D_3 und D_4 zusammensetzen, bleibt dem Modellierer überlassen, da dies vom Kontext der Prozesslandschaft abhängig ist.

- Des Weiteren können in der verfeinerten Prozesslandschaft zusätzliche Schnittstellen existieren. In diesem Fall existieren zusätzliche Zugriffe von den verfeinernden Aktivitäten a_i derart, dass mindestens eine Aktivität a_i lesend und eine Aktivität a_i' schreibend auf jede neue Schnittstelle zugreift (D_5, D_6). Auch hier ist die konkrete Auswahl der nichtleeren Teilmengen F_j und G_j, aus denen sich die Mengen D_5 respektive D_6 zusammensetzen, vom Kontext der Prozesslandschaft abhängig und bleibt daher dem Modellierer überlassen.

Die dritte Bedingung beschreibt die Auswirkungen auf den Aktivitätenbaum der Prozesslandschaft, der durch die Verfeinerung um mindestens zwei weitere Elemente (a, a_i) erweitert worden ist.

Die Einschränkung der Aktivitätsverfeinerung einer Prozesslandschaft auf die Menge der Blätter des zugehörigen Aktivitätenbaumes ist sinnvoll, da sie Überlegungen der Anbindung von Nachfolgern der ersetzten Aktivität a an die sie verfeinernden Aktivitäten a_i verhindert. Diese könnten ohne Wissen über inhaltliche Details der Prozesslandschaft leicht zu falschen Darstellungen führen. Es ist hervorzuheben, dass die Verfeinerung einer Prozesslandschaft bezüglich einer Aktivität für den Aktivitätenbaum eine echte Erweiterung bedeutet, da die verfeinerte Aktivität a in der Menge A erhalten bleibt.

Für das zugrundeliegende Petrinetz bedeutet diese Verfeinerung dagegen die Ersetzung von a durch ein neues Teilnetz, da hierfür lediglich die Menge $\{a_1, ..., a_n\} \subseteq leaves(AB')$ verwendet wird. Bezogen auf die grafische Darstellung des verfeinerten Petrinetzes ist die Aktivität a der Prozesslandschaft ω durch Elemente der Mengen $A'_{PN} = leaves(AB')$, S' und $Z'_{PN} = Z' \cap \big((leaves(AB') \times S') \cup (S' \times leaves(AB'))\big)$ ersetzt worden. Die Aktivität a wird dabei durch mindestens zwei andere Aktivitäten verfeinert, da $n \geq 2$. Dies ist sinnvoll, da bei einer Verfeinerung durch nur eine einzige Aktivität diese lediglich die Rolle der verfeinerten Aktivität übernehmen würde und der Informationsgehalt der Prozesslandschaft sich nicht geändert hätte.

Bemerkung 3.1.15. *Die hierarchischen Beziehungen zwischen Aktivitäten einer Prozesslandschaft ω können mit Hilfe der in Definition 3.1.8 formulierten Teilmengen wie folgt ausgedrückt werden:*

- *Die Menge $ref(AB)$ besteht aus allen inneren Knoten des Aktivitätenbaumes AB und damit allen verfeinerten Aktivitäten.*
- *Die Menge $leaves(AB)$ besteht aus allen Blättern des Aktivitätenbaumes AB und damit aus allen nicht weiter verfeinerten Aktivitäten der Prozesslandschaft ω.*
- *Die Menge $top(AB)$ besteht aus allen Aktivitäten der obersten Abstraktionsebene der Prozesslandschaft ω unterhalb der Quelle r.*
- *Die Quelle r des Aktivitätenbaumes AB ist ein Element der Menge A, das oberhalb der obersten Abstraktionsebene der Prozesslandschaft ω angeordnet ist.*

Notation 3.1.16. *Wenn keine Missverständnisse auftreten können, werden im Folgenden die Mengen $ref(AB)$, $leaves(AB)$ und $top(AB)$ kurz als Mengen ref, $leaves$ und top bezeichnet.*

Neben der Verfeinerung gibt es noch eine zweite Möglichkeit – die Erweiterung – zur Ergänzung einer Prozesslandschaft um weitere Informationen. Während bei der Verfeinerung bezüglich einer Aktivität $a \in A$ diese im zugrundeliegenden Petrinetz durch ein Teilnetz ersetzt und im zugehörigen Aktivitätenbaum eine Hierarchiestufe eingefügt wird, wird bei der Erweiterung das Petrinetz durch Hinzufügen von Kanten und Knoten (als Elemente der Menge $(A \cup S) \times (S \cup A)$) vergrößert. Alle Knoten des erweiterten Netzes bleiben erhalten.

Bezogen auf den Aktivitätenbaum einer Prozesslandschaft bewirkt eine Erweiterung die Vergrößerung der Nachfolgermenge für eine Aktivität $a \in A$. Dazu werden die hinzugefügten Aktivitäten eindeutig konkreten Elementen des Aktivitätenbaumes zugeordnet. Bildlich gesprochen wird jede „Aktivitätswaise“ von genau einer verfeinerten Aktivität „adoptiert“.

Definition 3.1.17. *Sei $\omega = (A, S, Z, AB)$ die Basis einer Prozesslandschaft und $A_{add} \subset A$ die Menge derjenigen Aktivitäten, die im Zuge einer Erweiterung der Prozesslandschaft hinzugefügt wurden. Die Menge aller Aktivitäten $a \in A$, deren Nachfolger Blätter sind, lässt sich formulieren als*

$$pred_{leaves} := \{a \in A \mid succ(a) \in leaves\}$$

Die Abbildung **adopt** *ordnet jeder Aktivität $a \in A_{add}$ eine übergeordnete Aktivität $a \in pred_{leaves}$ zu:*

$$adopt : A_{add} \rightarrow pred_{leaves}$$

Über die Abbildung $adopt$ wird der Aktivitätenbaum erweitert, indem die Anzahl der Blätter vergrößert wird. Die Tiefe der hierarchischen Struktur der korrespondierenden Prozesslandschaft wird hierbei nicht verändert.

Die Umkehrrelation von $adopt$ ist definiert als

$$adopt^{-1} := \{(a_1, a_2) \mid a_2 \in A_{add} \wedge a_1 = adopt(a_2)\}$$

Für sie gilt offensichtlich: $adopt^{-1} \subset AB$, da AB der Aktivitätenbaum der erweiterten Prozesslandschaft ist.

Die „Adoption“ erscheint auf den ersten Blick trivial. Sie ist jedoch für eine korrekte und vollständige Abbildung des Aktivitätenbaumes erforderlich, da es zwar für einen Modellierer aufgrund des Kontexts einer Prozesslandschaft deutlich wird, wo hinzugefügte Aktivitäten im korrespondierenden Aktivitätenbaum einzuordnen sind, diese Eindeutigkeit für die formale Basis jedoch nicht gegeben ist.

Schnittstellen spielen beim Process Landscaping ebenfalls eine zentrale Rolle. Eine Notation, die die Methode geeignet unterstützen soll, muss daher die Möglichkeit bieten, Schnittstellen als Elemente einer Prozesslandschaft sowohl auf oberster Abstraktionsebene als auch verfeinert auf unteren Ebenen darzustellen. Dies ist insbesondere für diejenigen Schnittstellen interessant, für die mindestens zwei verschiedenartige

Zugriffe von Aktivitäten existieren, d.h. für die eine Aktivität lesend und die andere schreibend auf die Schnittstelle zugreift.

Definition 3.1.18. *Sei $\omega = (A, S, Z, AB)$ die Basis einer Prozesslandschaft, $\mathcal{P}(S)$ die Potenzmenge von S und a_1, $a_2 \in A$, $a_1 \neq a_2$. Durch die Abbildung* **interface** *wird die Menge aller Schnittstellen zwischen zwei Aktivitäten a_1 und a_2 definiert:*

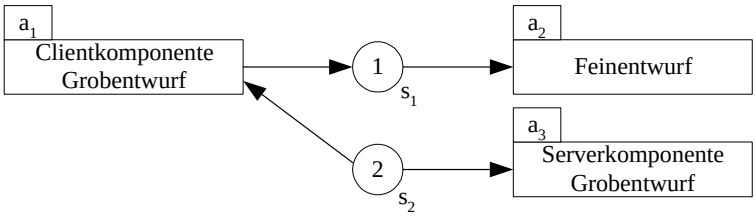
$$interface : A \times A \rightarrow \mathcal{P}(S) \qquad \textit{mit}$$

$$interface\big((a_1, a_2)\big) := \{s \in S | \big((a_1, s) \in Z \wedge (s, a_2) \in Z\big) \vee \big((a_2, s) \in Z \wedge (s, a_1) \in Z\big)\}$$

Bemerkung 3.1.19. *Um zu verdeutlichen, dass die Abbildung $interface$ auf Tupeln von Aktivitäten operiert, werden diese mit einem zusätzlichen Klammernpaar umschlossen.*

Beispiel:

Das in Abbildung 3.1 dargestellte Beispiel verdeutlicht grafisch, wann Schnittstellen Elemente einer Menge $interface\big((a_i, a_j)\big)$ mit $i, j \in \mathbb{N}$, $i \neq j$ sind.



1 Softwarearchitektur Clientkomponente
2 Auftrag Komponentenentwicklung

Abbildung 3.1: Schnittstellen zwischen Aktivitäten

$$\begin{aligned} \text{Es gilt: } & interface\big((a_1, a_2)\big) = \{s_1\}, \\ & interface\big((a_1, a_3)\big) = \emptyset \text{ und} \\ & interface\big((a_2, a_3)\big) = \emptyset. \end{aligned}$$

In Abbildung 3.1 ist die Schnittstelle s_1 Element von $interface\big((a_1, a_2)\big)$; s_2 gehört keiner der drei möglichen Mengen von Schnittstellen zwischen den abgebildeten Aktivitäten an.

Die nachfolgende Definition legt fest, wie sich die Verfeinerung einer Schnittstelle innerhalb einer Prozesslandschaft ω auf das zugrundeliegende Petrinetz auswirkt.

Definition 3.1.20. *Sei* $\omega = (A, S, Z, AB)$ *die Basis einer Prozesslandschaft und* $s \in S$*. Eine Prozesslandschaft*

$$\omega' = (A', S', Z', AB') \textit{ mit} \qquad S' = \big(S \setminus \{s\}\big) \cup \{s_1, ..., s_n\} \textit{ und } S \cap \{s_1, ..., s_n\} = \emptyset, n \geq 2$$

heißt ***Verfeinerung*** *einer Prozesslandschaft* ω ***bezüglich der Schnittstelle*** $s :\Leftrightarrow$

1. $A' = A$
2. *(a) Es existieren Mengen* E_1*,* E_2 *mit*

$$E_1 = \{(a, \tilde{s}) \mid \tilde{s} \in S \backslash s, a \in A, (a, \tilde{s}) \in Z\}$$
$$E_2 = \{(\tilde{s}, a) \mid \tilde{s} \in S \backslash s, a \in A, (\tilde{s}, a) \in Z\}$$

   (b) Des Weiteren existieren Mengen E_3*,* E_4 *mit*

$$E_3 = \{(a, s_i) \mid 1 \leq i \leq n, a \in A, (a, s) \in Z\}$$
$$E_4 = \{(s_i, a) \mid 1 \leq i \leq n, a \in A, (s, a) \in Z\}$$

   Es gilt: $Z' = \bigcup\limits_{1 \leq l \leq 4} E_l$

3. $AB' = AB$

In der Verfeinerung ω' ist die Schnittstelle s der Prozesslandschaft ω durch mindestens zwei Elemente der Menge S' ersetzt worden, die nicht in der Menge S liegen. Dies sichert die Steigerung des Informationsgehalts einer Prozesslandschaft, die durch die Verfeinerung beabsichtigt ist. Die erste Bedingung fordert, dass bei einer Verfeinerung einer Prozesslandschaft bezüglich einer Schnittstelle die Menge der Aktivitäten unverändert bleibt. Bedingung 2. legt die Menge der Zugriffe Z' als Vereinigung verschiedener Teilmengen fest:

- Analog zur Verfeinerung einer Prozesslandschaft bezüglich einer Aktivität gilt auch hier, dass alle Zugriffe, die bereits in Z existieren, aber nicht mit der Schnittstelle s verbunden waren, erhalten bleiben (E_1, E_2).
- Es kommen keine Zugriffe auf die Schnittstelle s mehr in ω' vor; stattdessen existieren Zugriffe auf alle verfeinernden Schnittstellen s_i, die diese in ω' mit denjenigen Aktivitäten verbinden, die in ω mit der jetzt verfeinerten Schnittstelle s verbunden waren (E_3, E_4). An dieser Stelle werden also wieder Elemente $z \in Z$ ausgetauscht.

Auf den Aktivitätenbaum einer Prozesslandschaft hat die Verfeinerung einer Schnittstelle keinerlei Auswirkungen, da dieser lediglich die Hierarchieinformationen für die Aktivitäten beinhaltet, dabei aber keine Zugriffe auf Schnittstellen berücksichtigt (Bedingung 3.).

Das der verfeinerten Basis einer Prozesslandschaft zugrundeliegende Petrinetz setzt sich hier aus den Mengen $A'_{PN} = A_{PN}$, S' und $Z'_{PN} = Z' \cap \big((leaves(AB) \times S') \cup (S' \times leaves(AB))\big)$ zusammen.

Bezogen auf das der Basis einer Prozesslandschaft zugrundeliegende Petrinetz ist ein wichtiger Unterschied zwischen der Verfeinerung bezüglich einer Aktivität a und bezüglich einer Schnittstelle s festzuhalten: Während bei der Verfeinerung einer Prozesslandschaft ω bezüglich einer Schnittstelle diese in der Menge $\mathcal{S}$ nicht mehr enthalten ist, wird bei einer Verfeinerung bezüglich einer Aktivität diese lediglich im zugehörigen Petrinetz der verfeinerten Prozesslandschaft ω' nicht mehr dargestellt. Die Aktivität a bleibt jedoch Element der Aktivitätenmenge und findet sich weiterhin im Aktivitätenbaum (vgl. Definition 3.1.14).

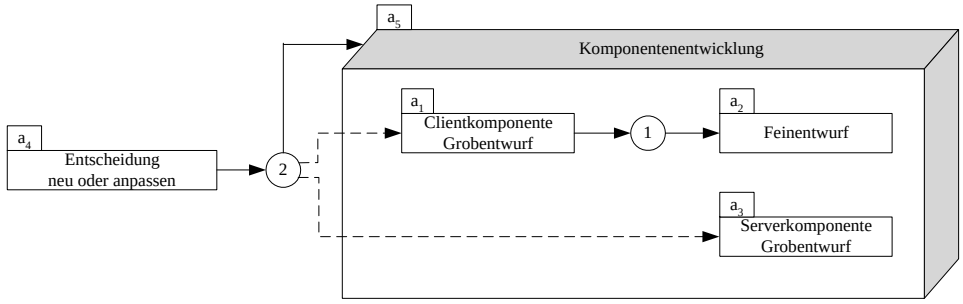
Mit den Definitionen 3.1.14 und 3.1.20 ist die Möglichkeit der Verfeinerung einer Prozesslandschaft bezüglich beider Grundmengen eines Petrinetzes (Aktivitäten und Schnittstellen) gegeben. Bei ihrer Durchführung muss jedoch zwingend eine sequentielle Reihenfolge eingehalten werden. Für einen Modellierer bedeutet dies, dass immer nur die zuletzt verfeinerte Prozesslandschaft weiter verfeinert werden darf. Die parallele Durchführung mehrerer Verfeinerungsschritte birgt die Gefahr von Inkonsistenzen, die nicht oder nur mit großem Aufwand korrigiert werden können. Die Ursache für die entstehenden Inkonsistenzen liegt in den modellierten Zugriffen zwischen parallel verfeinerten Aktivitäten und Schnittstellen, die in Abhängigkeit jedes einzelnen Verfeinerungsschrittes individuell anzupassen sind.

Die Forderung nach streng sequentieller Vorgehensweise bei Verfeinerungen unterscheidet das Process Landscaping von der traditionellen Art der Verfeinerung von Petrinetzen. Dort wird im Rahmen einer Verfeinerung eine Transition durch ein transitionsberandetes Teilnetz ersetzt bzw. eine Stelle durch ein stellenberandetes Teilnetz [Bau96], ohne eine mögliche parallele Vorgehensweise auszuschließen. Aus diesem Grund spricht man im Rahmen des Process Landscaping auch von der Verfeinerung einer Prozesslandschaft bezüglich einer Aktivität oder einer Schnittstelle anstatt von der Verfeinerung einer Aktivität oder Schnittstelle. Die Ursache für diesen Unterschied liegt vor allem darin, dass die Verfeinerung eines Petrinetzes, das einer Prozesslandschaft zugrundeliegt, immer auch Auswirkungen auf die Randstellen des zu ersetzenden Petrinetzelementes hat. Insbesondere müssen hier immer die Zugriffe, die mit dem ersetzten Petrinetzelement verbunden sind, angepasst werden und bestimmten Regeln genügen (vgl. Bedingungen 2. in den Definitionen 3.1.14 und 3.1.20).

Bezüglich der Schnittstellenverfeinerung ist hier noch ein weiterer wichtiger Unterschied festzuhalten: Beim Process Landscaping wird eine Schnittstelle nicht durch ein Teil- bzw. Subnetz ersetzt, sondern lediglich durch eine Menge von detaillierten Schnittstellen inklusive der Zugriffe ihrer Aktivitäten im Vor- und Nachbereich der verfeinerten Schnittstelle.

Bemerkung 3.1.21. *Wird eine Schnittstelle $s \in S$ verfeinert, die Element einer Menge $interface\big((a_1, a_2)\big)$ ist, spricht man auch von der Verfeinerung dieser Menge $interface\big((a_1, a_2)\big)$.*

Notation 3.1.22. *Sei $\omega = (A, S, Z, AB)$ die Basis einer Prozesslandschaft. Eine Menge $interface\big((a_1, a_2)\big)$ von Schnittstellen zwischen zwei Aktivitäten, von denen mindestens eine eine verfeinerte Aktivität ist (d.h. $a_1 \in ref$ oder $a_2 \in ref$), wird als **impliziit** bezeichnet. Ein Element einer solchen impliziten Schnittstellenmenge wird entsprechend **implizite Schnittstelle** genannt. Eine Menge von Schnittstellen zwischen zwei Blättern eines Aktivitätenbaumes wird als **explizit** bezeichnet; ihre Elemente werden entsprechend als **explizite Schnittstellen** bezeichnet.*



1 Softwarearchitektur Clientkomponente
2 Auftrag Komponentenentwicklung

Abbildung 3.2: Implizite und explizite Mengen von Schnittstellen zwischen Aktivitäten

Abbildung 3.2 verdeutlicht den Unterschied zwischen impliziten und expliziten Mengen von Schnittstellen. Die Menge $interface\big((\textit{Entscheidung neu oder anpassen}, \textit{Komponentenentwicklung})\big)$ mit der Schnittstelle *Auftrag Komponentenentwicklung* als Element ist implizit, da letztere Aktivität verfeinert ist. In Abbildung 3.2 ist dies durch die eingerahmten Aktivitäten dargestellt, die (neben weiteren) innerhalb der Aktivität *Komponentenentwicklung* stattfinden. Die Schnittstelle *Auftrag Komponentenentwicklung* ist gleichzeitig Element der expliziten Mengen $interface\big((\textit{Entscheidung neu oder anpassen}, \textit{Clientkomponente Grobentwurf})\big)$ und $interface\big((\textit{Entscheidung neu oder anpassen}, \textit{Serverkomponente Grobentwurf})\big)$. In Abbildung 3.2 ist dies durch die gestrichelten Pfeile angedeutet, die die Schnittstelle *Auftrag Komponentenentwicklung* mit den innerhalb der Aktivität *Komponentenentwicklung* dargestellten verfeinernden Aktivitäten verbindet. Es sei jedoch darauf hingewiesen, dass in einem Petrinetz, welches einer Prozesslandschaft ω zugrundeliegt, nur explizite Mengen von Schnittstellen vorkommen, da dort nur Aktivitäten der Menge $leaves$ auf die Schnittstellen zugreifen. Tabelle 3.1 fasst die Situation zusammen.

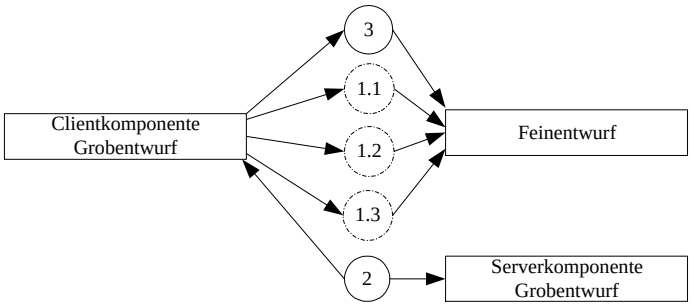
Bei Schnittstellen wird ebenfalls bewusst zwischen den Begriffen der Verfeinerung und der Erweiterung innerhalb einer Prozesslandschaft unterschieden (vgl. Verfeinerung und Erweiterung von Aktivitäten): Verfeinerung bedeutet die Ersetzung einer Schnittstelle durch mindestens zwei weitere. Dagegen wird bei einer Erweiterung mindestens

ein zusätzlicher Zugriff einer Aktivität auf eine Schnittstelle derart festgelegt, dass diese die Restriktionen für Elemente einer Menge $interface\big((a_1, a_2)\big)$ aus Definition 3.1.18 erfüllt.

Interface	**Landschaftselement**	**Schnittstelle**
$interface\big((a_1, a_2)\big)$	explizite Schnittstelle zwischen a_1, a_2	$\{s_1\}$
$interface\big((a_1, a_3)\big)$	explizite Schnittstelle zwischen a_1, a_3	$\emptyset$
$interface\big((a_4, a_1)\big)$	implizite Schnittstelle zwischen a_4, a_1	$\{s_2\}$
$interface\big((a_4, a_2)\big)$	explizite Schnittstelle zwischen a_4, a_2	$\emptyset$
$interface\big((a_4, a_3)\big)$	explizite Schnittstelle zwischen a_4, a_3	$\{s_2\}$

Tabelle 3.1: Zusammenfassung der in Abbildung 3.2 dargestellten Situation

Abbildung 3.3 verdeutlicht den Unterschied zwischen der Verfeinerung und der Erweiterung am Beispiel der beiden bereits in Abbildung 3.1 eingeführten Schnittstellen. Die mit gestrichelten Kreisen dargestellten Schnittstellen stellen eine Verfeinerung der ursprünglichen Schnittstelle *Softwarearchitektur Clientkomponente* und damit eine Verfeinerung der Menge $interface\big((\textit{Clientkomponente Grobentwurf}, \textit{Feinentwurf})\big)$ dar (vgl. Abbildung 3.1, Seite 66). Die Schnittstelle *Komponentenintegrationsplan* ist dagegen eine Erweiterung der Menge $interface\big((\textit{Clientkomponente Grobentwurf}, \textit{Feinentwurf})\big)$ und die Schnittstelle *Auftrag Komponentenentwicklung* eine Erweiterung der Menge $interface\big((\textit{Clientkomponente Grobentwurf}, \textit{Serverkomponente Grobentwurf})\big)$.



1.1 Grafische Beschreibung der Softwarearchitektur Clientkomponente
1.2 Verbale Beschreibung der Softwarearchitektur Clientkomponente, deutsch
1.3 Verbale Beschreibung der Softwarearchitektur Clientkomponente, englisch
2 Auftrag Komponentenentwicklung
3 Komponentenintegrationsplan

Abbildung 3.3: Verfeinerung und Erweiterung von Schnittstellen

Bemerkung 3.1.23. *Für alle nachfolgend betrachteten Prozesslandschaften wird vorausgesetzt, dass sie durch Verfeinerungen gemäß Definition 3.1.14 und Definition 3.1.20 und/oder Erweiterungen gemäß Definition 3.1.17 aus einer „Initialbasis“* $\omega_o = (\{a\}, \emptyset, \emptyset, \emptyset)$ *hervorgegangen sind.*

Die Voraussetzung ist wichtig, um für jede Prozesslandschaft eindeutig auf ihr Ausgangsmodell schließen zu können. Dieser Rückschluss wird in Abschnitt 3.2.1.4 noch detaillierter diskutiert (vgl. Definition 3.2.40).

Die bislang formulierten Definitionen sind ausreichend für die Modellierung von oberen Ebenen einer Prozesslandschaft und berücksichtigen gleichzeitig die für das Process Landscaping erforderlichen Konzepte zur Darstellung von Schnittstellen und Verfeinerungen. Lediglich die Einschränkung der auf den oberen Ebenen zu betrachtenden Informationen auf diejenigen, die zumeist als – unterscheidbare – Dokumenttypen produziert bzw. verarbeitet werden, ist bislang nicht explizit formuliert worden. Für diesen Zweck kann eine Prozesslandschaft leicht um eine Menge DT von Dokumenttypen und eine Abbildung erweitert werden, die jeder Schnittstelle genau einen Dokumenttyp zuordnet.

Für die Diskussion einer konkreten Prozesslandschaft mit beteiligten Personen (Mitarbeiter, Projektleiter, etc.) ist es sinnvoll, die Aktivitäten, Schnittstellen und Dokumenttypen mit sprechenden Namen zu versehen. Dies kann über eine Menge $NAME$ von Namen und einer Abbildung erfolgen, die jeder Aktivität, jeder Schnittstelle und jedem Dokumenttyp einen Namen zuordnet.

Schließlich kann auch die Anforderung des Process Landscaping nach der Berücksichtigung der geografischen Verteilung der Aktivitäten einer Prozesslandschaft realisiert werden über eine Menge O von Orten und einer Abbildung, die jeder Aktivität mindestens einen Ort zuordnet.

Die gerade angesprochenen Erweiterungen der Basis einer Prozesslandschaft führen zur ersten Ausbaustufe der Definition 3.1.11 (vgl. Seite 61):

Definition 3.1.24. *Sei $\omega = (A, S, Z, AB)$ die Basis einer Prozesslandschaft, DT eine Menge von Dokumenttypen mit $DT \neq \emptyset$, NAME die Menge aller Zeichenfolgen und $\mathcal{P}(O)$ die Potenzmenge der Menge aller Orte mit $O \neq \emptyset$. Sei weiterhin*

- $\mathbf{type}$ *eine Abbildung, die jeder Schnittstelle $s \in S$ genau einen Dokumenttypen $dt \in DT$ zuweist:*
$$type : S \rightarrow DT$$
- $\mathbf{name}$ *eine Abbildung, die allen Aktivitäten, Schnittstellen und Dokumenttypen der Prozesslandschaft PL_{top} einen Namen $n \in NAME$ zuordnet:*
$$name : A \cup S \cup DT \rightarrow NAME$$
- $\mathbf{local}$ *eine Abbildung, die jeder Aktivität $a \in A$ eine mindestens einelementige Menge von Orten $o \in O$ zuordnet:*
$$local : A \rightarrow \mathcal{P}(O) \backslash \{\emptyset\}$$

*Damit lässt sich die **erste Ausbaustufe einer Prozesslandschaft** definieren als ein Tupel*

$$PL_{top} := (\omega, DT, NAME, O, type, name, local)$$

Notation 3.1.25. *Wenn keine Missverständnisse auftreten können, wird eine Prozesslandschaft $PL_{top} = (\omega, DT, NAME, O, type, name, local)$ im Folgenden verkürzt als $PL_{top} = (A, S, Z, AB)$ bezeichnet.*

Bemerkung 3.1.26. *Der Index* top *der Prozesslandschaft PL_{top} deutet an, dass es sich um eine Prozesslandschaft handelt, deren obere Ebenen in der Petrinetz-Variante PPL (vgl. Einleitung zu Abschnitt 3.1, Seite 59) notiert sind.*

Mit der Abbildung $type$ kann für jede Schnittstelle der ersten Ausbaustufe einer Prozesslandschaft festgelegt werden, ob diese zum Beispiel Informationsobjekte vom Typ Anforderungsdokument, Entwurfsdokument, etc. enthält. Dies ist beispielsweise bei einer komponentenbasierten Softwareentwicklung sinnvoll, wenn verschiedene Komponenten von unterschiedlichen Entwicklergruppen realisiert werden und demzufolge für jede Komponente die obigen Dokumenttypen erstellt werden müssen. Die entstehenden Dokumente haben in diesem Fall jeweils den gleichen Typ, beziehen sich aber auf unterschiedliche Komponenten eines zu erstellenden Gesamtsystems.

Konsistenzbedingung 3.1.27. *Für eine eindeutige Identifizierung von Schnittstellen und Aktivitäten innerhalb von PL_{top} müssen diese einen eindeutigen Namen haben. Daher muss die Abbildung $name$ injektiv sein.*

Mit der Abbildung $local$ ist es möglich, den Verteilungsaspekt einer Prozesslandschaft zu berücksichtigen. Die Zuordnung mehrerer Orte zu einer Aktivität ist beispielsweise immer dann notwendig, wenn diese Aktivität als eine logische Einheit modelliert wurde, aber innerhalb eines realen Prozesses an verschiedenen Orten durchgeführt wird. Ein typisches Beispiel innerhalb der komponentenbasierten Softwareentwicklung ist die Aktivität *Component Engineering*, die mehrfach und meist an unterschiedlichen Orten innerhalb eines Projektes durchgeführt wird. Die Berücksichtigung der Anzahl der einer Aktivität zugeordneten Orte unterstützt unter anderem eine Analyse der Kommunikationsstruktur innerhalb einer Prozesslandschaft (vgl. Abschnitt 3.1.3).

Definition 3.1.28. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft. Bei einem Informationsaustausch zwischen Aktivitäten $a_1, a_2 \in A$, die an unterschiedlichen Orten stattfinden, lässt sich eine Abbildung definieren, die für die beteiligte Schnittstelle $s' \in S$ eine für sie gültige Ortemenge $local_s(s')$ zuordnet:*
Sei $s' \in interface\big((a_1, a_2)\big)$. Es gilt:

$$local_s : S \to \mathcal{P}(O)\backslash\{\emptyset\} \qquad \textit{mit} \qquad local_s(s') := local(a_1) \cup local(a_2)$$

Schnittstellen werden damit immer allen Orten zugeordnet, an denen auf sie zugreifende Aktivitäten durchgeführt werden.

Mit der ersten Ausbaustufe einer Prozesslandschaft können ihre oberen Ebenen unter Berücksichtigung aller hierfür diskutierten Anforderungen formuliert werden. Ihre grafische Repräsentation wird im nachfolgenden Abschnitt diskutiert. Dabei wird insbesondere auf Unterschiede zur grafischen Repräsentation konventioneller Petrinetze eingegangen.

3.1.2 Grafische Repräsentation einer Prozesslandschaft

Die Abbildungen 3.1 bis 3.3 auf den Seiten 66, 69 und 70 stellen Ausschnitte einer Prozesslandschaft zur komponentenbasierten Softwareentwicklung in Form von PLL-Petrinetzen dar. In Abbildung 3.2 ist dabei zusätzlich die Hierarchie zwischen Aktivitäten angedeutet. Für die grafische Repräsentation einer Prozesslandschaft in PLL-Notation sind neben diesen weitere Darstellungsformen entwickelt worden, die die Konzentration auf einzelne Aspekte der Landschaft auf unterschiedlichen Abstraktionsebenen unterstützen, indem sie auf unterschiedliche Art und Weise von Elementen der Prozesslandschaft abstrahieren. Diese Abstraktionen haben den Vorteil, dass man je nach Zielgruppe und Einsatzziel der grafischen Repräsentationen diese individuell unterstützen kann. Zielgruppe kann dabei das Management eines Unternehmens oder eine Gruppe der in den Softwareprozess involvierten Mitarbeiter sein, die auf jeweils unterschiedlichen Abstraktionsebenen über die dargestellten Prozesse diskutieren wollen. Als mögliche Einsatzziele sind die Dokumentation, aber auch die Analyse verschiedener Eigenschaften der modellierten Prozesse denkbar (vgl. Kapitel 2).

Dieser Abschnitt erläutert die grafischen Repräsentationsformen einer in PLL notierten Prozesslandschaft. Sie abstrahieren von der konventionellen Repräsentation eines Petrinetzes durch Stellen, Transitionen und Kanten. Die Anwendung der für eine Prozesslandschaft PL_{top} definierten Abbildung $name$ erleichtert die Diskussion über den Inhalt der verschiedenen Repräsentationen. Die Anwendung der Abbildungen $type$ und $local$ wird dagegen in den grafischen Darstellungen zunächst nicht weiter berücksichtigt, um diese nicht mit Informationen zu überladen.

Abbildung 3.4 zeigt die hierarchische Ordnung von Aktivitäten als Aktivitätenbaum am Beispiel einer Prozesslandschaft zur komponentenbasierten Softwareentwicklung. Wichtige inhaltliche Merkmale der abgebildeten Prozesslandschaft sind beispielsweise die Aktivitäten *Component Engineering* und *Domain Engineering*, die sich auf gleicher Abstraktionsebene befinden. Einige der Aktivitäten sind stärker verfeinert, bei anderen ist aus Gründen der Übersichtlichkeit von Details abstrahiert worden. Letztere sind in Abbildung 3.4 mit einem Sternchen gekennzeichnet. Die Aktivität *Komponentenentwicklung* innerhalb des Component Engineering ist zum Beispiel durch sieben weitere Aktivitäten verfeinert, die Aktivität *Konfigurationsmanagement (CE)* durch drei weitere. Die Abkürzung (AE) zeigt an, dass die damit versehenen Aktivitäten dem Application Engineering zugeordnet sind. Gleichnamige Aktivitäten – bis auf das Kürzel – existieren auch innerhalb des Component Engineering. Dort sind sie entsprechend mit dem Kürzel (CE) versehen worden.

In dieser Darstellungsform werden weder Schnittstellen und noch Zugriffe betrachtet. Sie sind ausgeblendet, um sich auf die Hierarchieinformationen konzentrieren zu können, die in der grafischen Repräsentation eines Petrinetzes nicht erkennbar sind. Ein Aktivitätenbaum repräsentiert damit immer nur einen Teil einer Prozesslandschaft. Er lässt sich nicht als Petrinetz darstellen. Der in Abbildung 3.4 gezeigte Ausschnitt der Prozesslandschaft entspricht den in Definition 3.1.11 eingeführten Mengen A (exklusive der über das Sternsymbol abstrahierten Aktivitäten) und AB aus PL_{top} mit

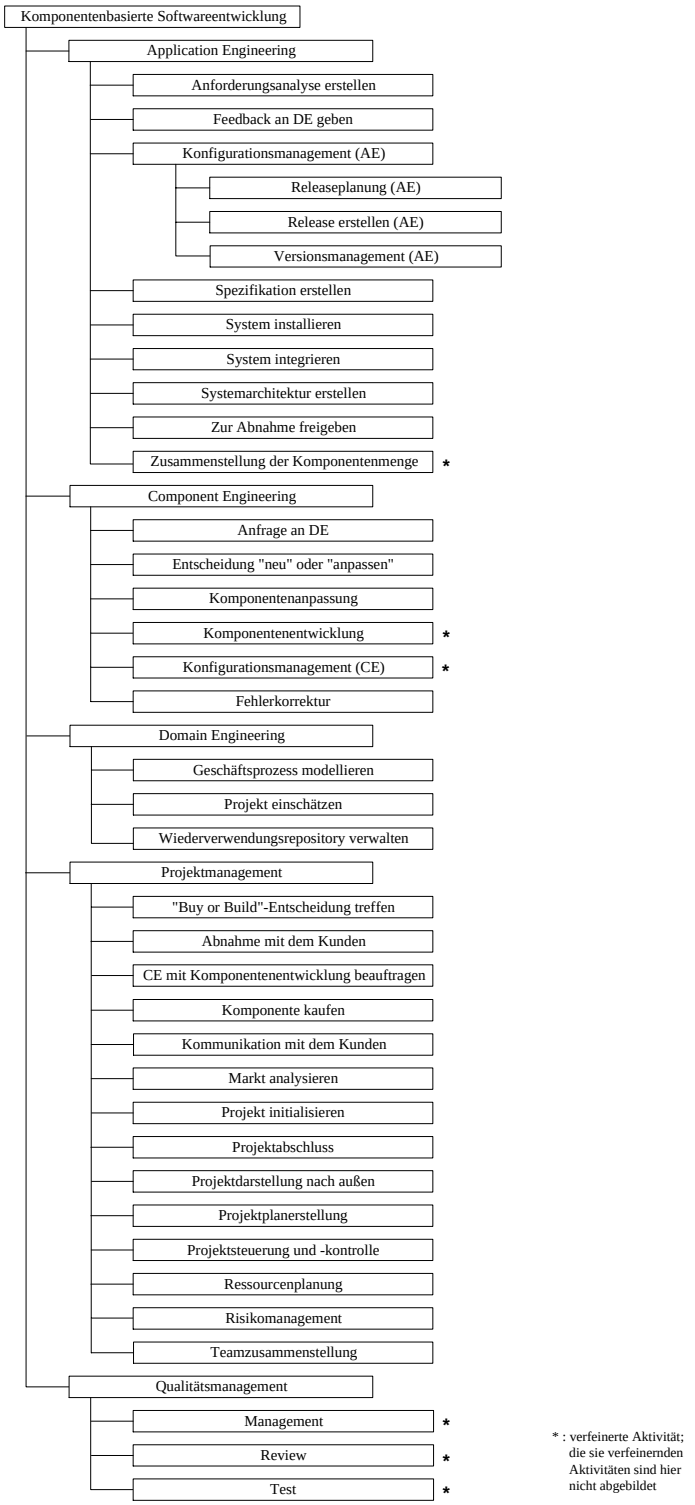


Abbildung 3.4: Aktivitätenbaum einer Prozesslandschaft zur komponentenbasierten Softwareentwicklung

$A =$
{*Komponentenbasierte Softwareentwicklung, Application Engineering, Anforderungsanalyse erstellen, Feedback an DE geben, Konfigurationsmanagement (AE), Releaseplanung (AE), Release erstellen (AE), Versionsmanagement (AE), Spezifikation erstellen, System installieren, System integrieren, Systemarchitektur erstellen, Zur Abnahme freigeben, Zusammenstellung der Komponentenmenge, Component Engineering, Anfrage an DE, Entscheidung „neu“ oder „anpassen“, Komponentenanpassung, Komponentenentwicklung, Konfigurationsmanagement (CE), Fehlerkorrektur, Domain Engineering, Geschäftsprozess modellieren, Projekt einschätzen, Wiederverwendungsrepository verwalten, Projektmanagement, „Buy or Build“-Entscheidung treffen, Abnahme mit dem Kunden, CE mit Komponentenentwicklung beauftragen, Komponente kaufen, Kommunikation mit dem Kunden, Markt analysieren, Projekt initialisieren, Projektabschluss, Projektdarstellung nach außen, Projektplanerstellung, Projektsteuerung und -kontrolle, Ressourcenplanung, Risikomanagement, Teamzusammenstellung, Qualitätsmanagement, Management, Review, Test*}

$AB =$
{ *(Komponentenbasierte Softwareentwicklung, Application Engineering), (Komponentenbasierte Softwareentwicklung, Component Engineering), (Komponentenbasierte Softwareentwicklung, Domain Engineering), (Komponentenbasierte Softwareentwicklung, Projektmanagement), (Komponentenbasierte Softwareentwicklung, Qualitätsmanagement),*
(Application Engineering, Anforderungsanalyse erstellen), (Application Engineering, Feedback an DE geben), (Application Engineering, Konfigurationsmanagement (AE)), (Application Engineering, Releaseplanung (AE)), (Application Engineering, Release erstellen (AE)), (Application Engineering, Versionsmanagement (AE)), (Konfigurationsmanagement (AE), Releaseplanung (AE)), (Konfigurationsmanagement (AE), Release erstellen (AE)), (Konfigurationsmanagement (AE), Versionsmanagement (AE)), (Application Engineering, Spezifikation erstellen), (Application Engineering, System installieren), (Application Engineering, System integrieren), (Application Engineering, Systemarchitektur erstellen), (Application Engineering, Zur Abnahme freigeben), (Application Engineering, Zusammenstellung der Komponentenmenge),
(Component Engineering, Anfrage an DE), (Component Engineering, Entscheidung „neu“ oder „anpassen“), (Component Engineering, Komponentenanpassung), (Component Engineering, Komponentenentwicklung), (Component Engineering, Konfigurationsmanagement (CE)), (Component Engineering, Fehlerkorrektur),
(Domain Engineering, Geschäftsprozess modellieren), (Domain Engineering, Projekt einschätzen), (Domain Engineering, Wiederverwendungsrepository verwalten),
(Projektmanagement, „Buy or Build“-Entscheidung treffen), (Projektmanagement, Abnahme mit dem Kunden), (Projektmanagement, CE mit Komponentenentwicklung beauftragen), (Projektmanagement, Komponente kaufen), (Projektmanagement, Kommunikation mit dem Kunden), (Projektmanagement, Markt analysieren), (Projektmanagement, Projekt initialisieren), (Projektmanagement, Projektabschluss), (Projektmanagement, Projektdarstellung nach außen), (Projektmanagement, Projektplaner-

stellung), (Projektmanagement, Projektsteuerung und -kontrolle), (Projektmanagement, Ressourcenplanung), (Projektmanagement, Risikomanagement), (Projektmanagement, Teamzusammenstellung),
(Qualitätsmanagement, Management), (Qualitätsmanagement, Review), (Qualitätsmanagement, Test) $\}$.

Die in die textuelle Beschreibung der Menge AB eingefügten Absätze entsprechen den Aktivitäten der obersten Landschaftsebene und sollen die Vergleichbarkeit mit Abbildung 3.4 erleichtern. Insgesamt beinhaltet die in Abbildung 3.4 dargestellte Prozesslandschaft zur komponentenbasierten Softwareentwicklung 65 Aktivitäten (inklusive der dort lediglich mit einem Sternchen angedeuteten), davon 53 nicht weiter verfeinerte.

Die Aktivität *Application Engineering* ist in Abbildung 3.5 verfeinert dargestellt. Dass diese Abbildung tatsächlich einer bzw. genau der gerade erwähnten Verfeinerung entspricht, lässt sich jedoch nur mit Hilfe des Aktivitätenbaumes (vgl. Abbildung 3.4) erkennen, der belegt, dass alle verfeinernden Aktivitäten im Aktivitätenbaum AB die Aktivität *Application Engineering* als gemeinsamen direkten Vorgänger haben. Abbildung 3.5 zeigt zusätzlich die Existenz von Schnittstellen zwischen Aktivitäten, angedeutet durch bidirektionale Pfeile. Diese Form der grafischen Repräsentation einer Prozesslandschaft entspricht ebenfalls nicht der Darstellung durch ein konventionelles Petrinetz, da die Schnittstellen nicht explizit dargestellt sind und somit eine Grundmenge eines Petrinetzes nicht berücksichtigt ist.

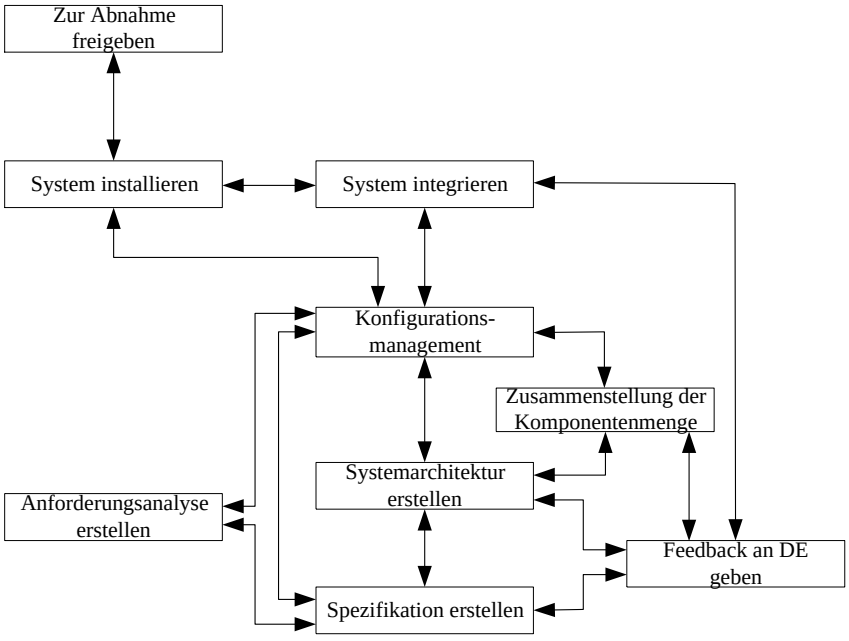
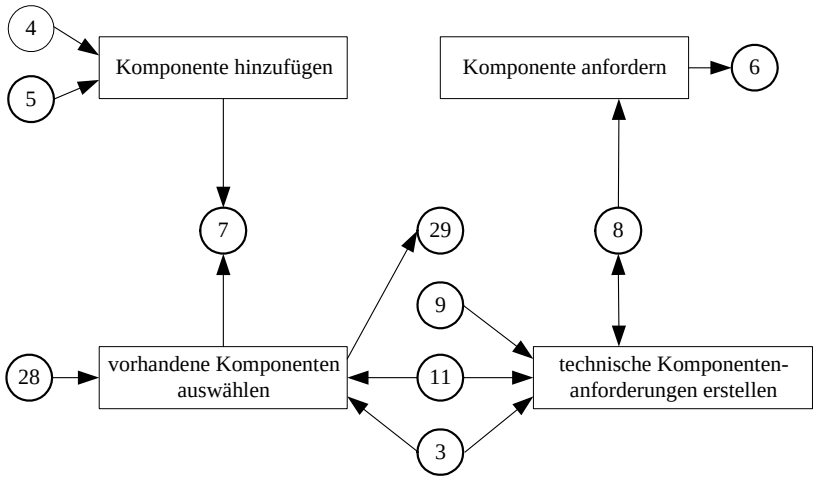


Abbildung 3.5: Verfeinerung der Aktivität *Application Engineering* mit angedeuteten Schnittstellen

Anhand des in Abbildung 3.5 dargestellten Ausschnitts der Beispielprozesslandschaft kann ebenfalls keine vollständige Beschreibung aller Mengen des Tupels (A, S, Z, AB) angegeben werden. Es kann lediglich eine Teilmenge $A' \subset A$ angegeben werden. Die bidirektionalen Pfeile lassen erkennen, dass S und Z nichtleere Mengen sind, ihre konkreten Elemente sind im dargestellten Ausschnitt nicht aufgeführt. Über den Aktivitätenbaum AB kann ebenfalls keine Aussage getroffen werden.

Sollen Details über die im realen Prozess auszutauschenden Informationsobjekte dargestellt werden, können die bidirektionalen Pfeile durch die Angabe konkreter Schnittstellen verfeinert werden. In diesem Fall wird eine Darstellungsform verwendet, die in Abbildung 3.6 am Beispiel der Verfeinerung der Aktivität *Zusammenstellung der Komponentenmenge* verdeutlicht wird. Dass es sich auch bei diesem Ausschnitt der Prozesslandschaft um eine Verfeinerung einer Aktivität $a \in A$ handelt, ist ebenfalls nur mit Hilfe des Aktivitätenbaumes AB aus Abbildung 3.4 erkennbar.



3 Anforderungsdokument
4 getestete Komponente
5 gekaufte Komponente
6 Komponentenanforderung
7 projektbezogene Komponentenmenge
8 technische Komponentenanforderung
9 Review technische Komponentenanforderung
11 Systemarchitektur
28 Feedback von Repository
29 Anfrage an Repository

Abbildung 3.6: Verfeinerung der Aktivität *Zusammenstellung der Komponentenmenge*

In Abbildung 3.6 sind vier Aktivitäten und zehn Schnittstellen dargestellt. Letztere werden meist ähnlich wie ihre Dokumenttypen benannt. Beispielsweise wird die technische Komponentenanforderung – als Anforderungsdokument typisiert (was jedoch in der Abbildung nicht ersichtlich ist) – von der sie erstellenden Aktivität zur Aktivität *Komponente anfordern* weitergeleitet. Für den in Abbildung 3.6 dargestellten Ausschnitt der Beispielprozesslandschaft können zu folgenden Mengen Aussagen bezüglich ihrer Elemente gemacht werden:

- $A \supset A''$ und $A'' =$
  $\{$ *Komponente hinzufügen, Komponente anfordern, vorhandene Komponenten auswählen, technische Komponentenanforderungen erstellen*$\}$
- $S \supset S''$ und $S'' =$
  $\{$ *Anforderungsdokument, getestete Komponente, gekaufte Komponente, Komponentenanforderung, projektbezogene Komponentenmenge, technische Komponentenanforderung, Review technische Komponentenanforderung, Systemarchitektur, Feedback von Repository, Anfrage an Repository*$\}$
- $Z \supset Z''$ und $Z'' =$
  $\{$ *(getestete Komponente, Komponente hinzufügen), (gekaufte Komponente, Komponente hinzufügen), (Komponente hinzufügen, projektbezogene Komponentenmenge), (Feedback von Repository, vorhandene Komponenten auswählen), (vorhandene Komponenten auswählen, projektbezogene Komponentenmenge), (vorhandene Komponenten auswählen, Anfrage an Repository), (Systemarchitektur, vorhandene Komponenten auswählen), (Anforderungsdokument, vorhandene Komponenten auswählen), (Anforderungsdokument, technische Komponentenanforderungen erstellen), (Systemarchitektur, technische Komponentenanforderungen erstellen), (Review technische Komponentenanforderung, technische Komponentenanforderungen erstellen), (technische Komponentenanforderung, technische Komponentenanforderungen erstellen), (technische Komponentenanforderungen erstellen, technische Komponentenanforderung), (technische Komponentenanforderung, Komponente anfordern), (Komponente anfordern, Komponentenanforderung)*$\}$

Der Unterschied zur konventionellen Darstellung eines Petrinetzes liegt in Abbildung 3.6 in der Andeutung von Datenflüssen zu weiteren Aktivitäten, die außerhalb der abgebildeten Verfeinerung liegen. Alle in der Abbildung fett dargestellten Schnittstellen der Prozesslandschaft repräsentieren Schnittstellen auch zu Aktivitäten, die nicht zur Verfeinerung der Aktivität *Zusammenstellung der Komponentenmenge* gehören und daher auf der abgebildeten Abstraktionsebene nicht sichtbar sind. Auf die Schnittstelle *Systemarchitektur* wird zum Beispiel von der Aktivität *Systemarchitektur erstellen* lesend und schreibend zugegriffen. Diese Aktivität befindet sich jedoch in einem anderen Teilbaum des zugehörigen Aktivitätenbaumes (vgl. Abbildung 3.4).

Neben den eher aktivitätsbezogenen Darstellungsformen existiert die Dokumentensicht als eine weitere Darstellungsform. Sie unterstützt einen schnellen Überblick auf die Zugriffsart (lesend oder schreibend) und die Anzahl der Zugriffe auf konkrete Schnittstellen. Abbildung 3.7 zeigt die Dokumentensicht auf die Schnittstelle *Systemarchitektur*. Dabei wird sowohl von der Einordnung der zugreifenden Aktivitäten in die hierarchische Struktur als auch von der Reihenfolge ihrer Zugriffe abstrahiert. Trotzdem kommt diese Darstellungsform der konventionellen Repräsentation eines Petrinetzes am nächsten, da sowohl Schnittstellen und Aktivitäten als auch Zugriffe und damit Datenflüsse eindeutig identifiziert werden können. Lediglich der Kontrollfluss ist nicht erkennbar.

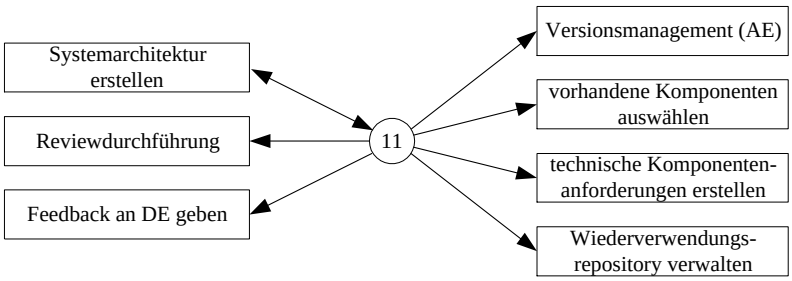


Abbildung 3.7: Dokumentensicht auf die Schnittstelle *Systemarchitektur*

Die in Abbildung 3.7 dargestellten Elemente stellen einen weiteren Ausschnitt der Beispielprozesslandschaft dar. Auch hier können die Elemente von Teilmengen der Mengen A, S und Z konkret aufgeführt werden:

- $A \supset A'''$ und $A''' =$
  $\{$*Systemarchitektur erstellen, Reviewdurchführung, Feedback an DE geben, Versionsmanagement (AE), vorhandene Komponenten auswählen, technische Komponentenanforderungen erstellen, Wiederverwendungsrepository verwalten*$\}$
- $S \supset S'''$ und $S''' =$
  $\{$*Systemarchitektur*$\}$
- $Z \subset Z'''$ und $Z''' =$
  $\{$*(Systemarchitektur erstellen, Systemarchitektur), (Systemarchitektur, Systemarchitektur erstellen), (Systemarchitektur, Reviewdurchführung), (Systemarchitektur, Feedback an DE geben), (Systemarchitektur, Versionsmanagement (AE)), (Systemarchitektur, vorhandene Komponenten auswählen), (Systemarchitektur, technische Komponentenanforderungen erstellen), (Systemarchitektur, Wiederverwendungsrepository verwalten)*$\}$

Zusammenfassend kann zwischen vier verschiedenen Darstellungsformen einer Prozesslandschaft (bzw. Ausschnitten davon) in PLL-Notation unterschieden werden:

- dem Aktivitätenbaum,
- zwei Arten der Darstellung von Aktivitätsverfeinerungen (mit und ohne Berücksichtigung von Schnittstellen) und
- der Dokumentensicht.

Die erste Darstellungsform konzentriert sich – als einzige – auf die hierarchische Struktur der Aktivitäten einer Prozesslandschaft, die nächsten beiden auf die Relationen zwischen Aktivitäten, die jeweils auf gleicher Abstraktionsebene angeordnet sind,

und die letzte stellt die Zugriffe auf eine konkrete Schnittstelle dar. Ein weiterer wichtiger Punkt ist die Tatsache, dass die grafische zweidimensionale Darstellung einer Prozesslandschaft durch ein Petrinetz niemals alle Elemente der Aktivitätenmenge A darstellt, sondern immer nur eine Teilmenge der Elemente des zugehörigen Aktivitätenbaumes AB, konkret die Blätter $leaves(AB)$ (vgl. Definition 3.1.11, Seite 61). Eine vollständige Darstellung einer Prozesslandschaft lässt sich also nur mit der Kombination Petrinetz/Aktivitätenbaum erreichen.

Die verschiedenen Darstellungsformen werden unter anderem in Abschnitt 3.1.3 verwendet, in dem die Definitionen für die Analyse konkreter Kommunikationseigenschaften mit Beispielen unterlegt sind.

3.1.3 Erweiterungen zur Analyse statischer Kommunikationseigenschaften

In diesem Abschnitt wird die in Abschnitt 3.1.1 eingeführte Petrinetz-Variante PLL, die die Modellierung der oberen Ebenen einer Prozesslandschaft gemäß der Process Landscaping Methode unterstützt, um Möglichkeiten der Analyse dieser oberen Ebenen erweitert. Der Fokus liegt dabei auf Analysemöglichkeiten von Kommunikationseigenschaften einer verteilten Prozesslandschaft und damit auf der modellierten Semantik der Prozesse und ihrer Kommunikation untereinander. In PLL wird die Formulierung dieser Kommunikationseigenschaften über die Definition entsprechender Funktionen realisiert, wobei die Funktionswerte konkrete Ausprägungen von Eigenschaften der Landschaftselemente repräsentieren. Die zusätzliche Auszeichnung spezifischer Teilnetze als Elemente einer Kommunikationsinfrastruktur unterstützt die Analyse der Kommunikation in einer verteilten Prozesslandschaft ebenfalls.

Bemerkung 3.1.29. *Alle Eigenschaften, die nachfolgend für Petrinetze definiert werden, gelten gleichzeitig auch für Prozesslandschaften, die durch ein entsprechendes Petrinetz repräsentiert werden. Beispielsweise heißt eine Prozesslandschaft – unabhängig von ihrer Ausbaustufe – kohärent, wenn sie durch ein kohärentes Petrinetz repräsentiert wird.*

Für die Analyse verschiedener Abstraktionsebenen einer in PLL modellierten Prozesslandschaft ist es sinnvoll, zwischen Zugriffen von verfeinerten Aktivitäten auf Schnittstellen und denen von nicht weiter verfeinerten Aktivitäten auf Schnittstellen zu unterscheiden. Diese Unterscheidung erleichtert es dem Analysten, die für seine Analyse relevante Teilmenge an Zugriffen gezielter zu betrachten. Soll beispielsweise die Kommunikationsstruktur der unteren Abstraktionsebene einer in PLL modellierten Prozesslandschaft untersucht werden, reicht die Berücksichtigung derjenigen Zugriffe aus, die von nicht weiter verfeinerten Aktivitäten ausgehen. Soll dagegen die Kommunikationsstruktur einer gesamten Prozesslandschaft untersucht werden, ergibt sich diese aus der Menge der kleinteiligen Kommunikationsstrukturen auf den unteren Ebenen.

Definition 3.1.30. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft. Ein Zugriff von einer Aktivität $a \in A$ auf eine Schnittstelle $s \in S$ heißt **explizit** genau dann, wenn die Aktivität a nicht verfeinert ist und auf die Schnittstelle s lesend oder schreibend zugreift, also wenn*

$$(a, s) \in Z \vee (s, a) \in Z \text{ mit } a \in leaves.$$

Die Menge

$$explicit := \{(a, s) \in Z \mid a \in leaves\} \cup \{(s, a) \in Z \mid a \in leaves\}$$

enthält alle expliziten Zugriffe einer Prozesslandschaft.

Die Menge $explicit$ umfasst damit die Menge aller Zugriffe zwischen Schnittstellen und Blättern des Aktivitätenbaumes AB.

Definition 3.1.31. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft. Ein Zugriff von einer Aktivität $a \in A$ auf eine Schnittstelle $s \in S$ heißt **implizit** genau dann, wenn es einen Nachfolger $b \in succ(a)$ der Aktivität a gibt, der auf die Schnittstelle s lesend oder schreibend zugreift.*
Die Menge

$$\begin{aligned} implicit :=& \{(a, s) \in Z \mid \exists\, b \in succ(a) : (b, s) \in Z\} \cup \\ & \{(s, a) \in Z \mid \exists\, b \in succ(a) : (s, b) \in Z\} \end{aligned}$$

enthält alle impliziten Zugriffe einer Prozesslandschaft.

Die Menge $implicit$ umfasst damit die Menge aller Zugriffe von verfeinerten Aktivitäten auf Schnittstellen.
In Abbildung 3.2 auf Seite 69 stellt beispielsweise der lesende Zugriff der Aktivität *Komponentenentwicklung* auf die Schnittstelle *Auftrag Komponentenentwicklung* einen impliziten Zugriff dar, während der lesende Zugriff der Aktivität *Serverkomponente Grobentwurf* auf diese Schnittstelle ein expliziter Zugriff ist.
Zur Unterstützung eines Modellierers bzw. eines Modellierungsteams, welches gemeinsam eine komplexe Prozesslandschaft erstellt, wird der Begriff der Vollständigkeit eingeführt. Dieser hilft, offensichtliche Modellierungslücken in einer Prozesslandschaft aufzudecken, die spätere Analyseergebnisse verfälschen können.

Definition 3.1.32. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft.*
*PL_{top} heißt **vollständig**:$\Leftrightarrow$*

1. $\forall\, a \in leaves\, \exists\, s \in S : (a, s) \in Z \vee (s, a) \in Z$
2. $\forall\, s \in S\, \exists\, a \in A : (a, s) \in Z \vee (s, a) \in Z$
3. $\forall\, a \in A : |succ(a)| \neq 1$

Die Forderung nach der Vollständigkeit einer Prozesslandschaft PL_{top} verhindert durch die ersten beiden Bedingungen die Modellierung von isolierten Aktivitäten und Schnittstellen. Die dritte Bedingung fordert die Verfeinerung von Aktivitäten durch mindestens zwei weitere. Dies ist sinnvoll, da bei einer Verfeinerung durch nur eine einzige Aktivität diese lediglich die Rolle der verfeinerten Aktivität übernehmen würde und der Informationsgehalt der Prozesslandschaft sich damit nicht geändert hätte. Eine solche Situation kann den Modellierer auf die Unvollständigkeit seiner Arbeit hinweisen.

Im Rahmen der Erweiterungen von PLL, die speziell für die Analyse statischer Kommunikationseigenschaften erforderlich sind, werden nachfolgend Attribute inklusive ihrer Wertemenge für Zugriffe $z \in Z$ definiert. Konkret sind dies die Attribute *Persistenz, Synchronität, Veränderbarkeit, Privatheit, Verschlüsselung* und *Mehrfachversand*. Mit ihrer Hilfe werden Eigenschaften einer Prozesslandschaft formuliert, die sich aus der syntaktischen Struktur des zugrundeliegenden Petrinetzes nicht ableiten lassen.

Definition 3.1.33. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft und PL_{top} vollständig. Die Menge der möglichen Attributwerte von Zugriffen $z \in Z$ ist definiert als*

$$\mathbf{switch} := \{1, 0, \mathit{undefined}\}$$

Notation 3.1.34. *Die Elemente der Menge $switch$ heißen **Schalterwerte**.*

Die Elemente eines Zugriffs $z = (a, s) \in Z \vee z = (s, a) \in Z$ können durch Abbildungen mit verschiedenen Attributen versehen werden, die jeweils Attributwerte der Menge $switch$ annehmen. Zu Beginn der Modellierung einer Prozesslandschaft sind die konkreten Schalterwerte oft noch nicht bekannt, daher ist der Standardwert $undefined$. Er ermöglicht ein späteres und sukzessives Setzen aller Schalterwerte auf 0 oder 1. Ihre Bedeutung hängt von den jeweiligen Abbildungen ab und wird jeweils am Beispiel erläutert.

Definition 3.1.35. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft und PL_{top} vollständig. Die Abbildungen* $\mathbf{per}$, $\mathbf{synch}$, $\mathbf{change}$, $\mathbf{coded}$, $\mathbf{priv}$ *und* $\mathbf{mult}$ *legen für alle Zugriffe $z \in Z$ einer Prozesslandschaft deren Ausprägungen bezüglich Persistenz, Synchronität, Veränderbarkeit, Verschlüsselung, Privatheit und Mehrfachversand eines Informationsaustausches fest:*

Persistenz:	$per : Z \rightarrow switch$
Synchronität:	$synch : Z \rightarrow switch$
Veränderbarkeit:	$change : Z \rightarrow switch$
Verschlüsselung:	$coded : Z \rightarrow switch$
Privatheit:	$priv : Z \rightarrow switch$
Mehrfachversand:	$mult : Z \rightarrow switch$

Beispiel:
Wird der Softwarecode verschiedener Komponenten sowohl bei der erzeugenden Aktivität und der sie verwendenden Aktivität als auch in einem Wiederverwendungsrepository lokal gespeichert, so gilt für alle korrespondierenden Zugriffe $z \in Z$, dass $per(z) = 1$. Für diese sollte in Betracht gezogen werden, eine gemeinsame Datenbasis zu schaffen, um Mehrfacharchivierung und somit mögliche Inkonsistenzen durch Redundanzen zu vermeiden. Um solche Inkonsistenzen zu verhindern, müssten andernfalls alle archivierenden Aktivitäten über Änderungen an Dokumenten informiert werden. Ein durchdachter Aktualisierungsmechanismus würde notwendig werden.

Die Erwartung einer synchronen bzw. asynchronen Kommunikation hat Auswirkungen auf die dafür notwendige Infrastruktur: Briefpost kann ausschließlich als asynchrone Kommunikation modelliert werden ($synch(z) = 0$), während Telefonate oder (Video-) Konferenzen immer als synchron zu modellieren sind ($synch(z) = 1$). Wird eine Kommunikation als synchron eingestuft, erfolgt keine temporäre Zwischenspeicherung der auszutauschenden Daten in Wartebuffern, jedoch müssen sich beide Aktivitäten zeitlich aufeinander abstimmen.

Der Inhalt einer Datei im pdf-Format ist als nicht veränderbar ($change(z) = 0$) einzustufen. Eine MS-Word-Datei kann dagegen von jedem Empfänger verändert werden ($change(z) = 1$), sofern er ein entsprechendes Textverarbeitungssystem zur Verfügung hat. Signierte Dokumente sollten als nicht veränderbar attributiert werden. Veränderbarkeit ist immer dann zu verhindern, wenn Urheberrechte gewahrt werden sollen.

Verschlüsselung von Kommunikation ist seit jeher ein grundlegendes Problem. Zum einen ist bei einer Kommunikation häufig sicherzustellen, dass sie nicht von Dritten abgehört wird ($coded(z) = 1$). Zum anderen verlangsamt die Verschlüsselung die Kommunikationsgeschwindigkeit und stellt teilweise hohe Ansprüche an die verwendete Technik. Hier ist immer ein vertretbarer Mittelweg zu finden.

Beispiele für private Kommunikation sind Briefpost, E-Mail oder Telefon ($priv(z) = 1$). Beispiele für öffentliche Kommunikation ($priv(z) = 0$) sind Aushänge, Internet-Newsgroups oder Werbeanzeigen.

Mehrfachversand ($mult(z) = 1$) gibt es sowohl bei der herkömmlichen Kommunikation (Serienbriefe oder Anzeigen in Zeitungen/Aushängen) als auch bei der elektronischen Kommunikation (E-Mail-Verteiler, Internet-Newsgroups oder Webseiten).

Notation 3.1.36. *Alle durch die Abbildungen* $per, synch, coded, change, priv$ *und* $mult$ *repräsentierten Attribute einer Prozesslandschaft heißen* ***Kommunikationsattribute****.*

Nach der Definition der Kommunikationsattribute wird im Folgenden festgelegt, wie sie in einer Art Baukastensystem zusammengesetzt werden können, um die Kommunikationsstruktur einer Prozesslandschaft analysieren zu können. Diese kann mit Hilfe von Kommunikationskanälen beschrieben werden, die jeweils zwei Aktivitäten a_1 und a_2 und den Elementen der zugehörigen Mengen $interface\big((a_1, a_2)\big)$ zugeordnet sind und als besondere Teilnetze innerhalb des zugrundeliegenden Petrinetzes ausgezeich-

net werden. Kommunikationskanäle oder kurz Kanäle beschreiben durch die Ausprägung der zugehörigen Kommunikationsattribute die Eigenschaften der Kommunikation zwischen zwei Aktivitäten. Dabei erfüllt eine auf eine Schnittstelle schreibend zugreifende Aktivität a_1 die Rolle eines Senders und eine lesend zugreifende Aktivität a_2 die Rolle eines Empfängers.

Definition 3.1.37. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft und PL_{top} vollständig. Sei weiterhin $CC \neq \emptyset$ eine Menge, deren Elemente Kommunikationskanäle repräsentieren und $ZZ \subset Z \times Z$, wobei für alle $a_1, a_2 \in A$ mit $a_1 \neq a_2$, $s \in S$ und $(z_1, z_2) \in ZZ$ gilt:*

1. $z_1 = (a_1, s) \Rightarrow z_2 = (s, a_2)$
2. $z_1 = (s, a_1) \Rightarrow z_2 = (a_2, s)$

Die Abbildung **channel** *ordnet jedem Tupel $(z_1, z_2) \in ZZ$ genau einen Kommunikationskanal zu:*

$$channel : ZZ \rightarrow CC$$

Damit stellen z_1 und z_2 Zugriffe zur gleichen Schnittstelle s dar, wobei einer der beiden als Schreibzugriff definiert ist, der jeweils andere als Lesezugriff. Diese Restriktionen für die Urbildmenge der Abbildung $channel$ sind notwendige Voraussetzung für einen funktionsfähigen Informationsaustausch.

Beispiel:

Abbildung 3.8 zeigt beispielhaft verschiedene Kommunikationskanäle innerhalb einer Softwareprozesslandschaft. Die Abkürzung (AE) zeigt an, dass die beiden damit versehenen Aktivitäten verfeinernde Aktivitäten des *Application Engineering* sind.

Die Abbildung enthält insgesamt sieben Zugriffe, wobei (z_1, z_3), (z_1, z_4), (z_1, z_5), (z_1, z_6) und (z_1, z_7) fünf mögliche Kommunikationskanäle darstellen. Ein Modellierer kann die Auswahl der Kanäle aber auf (z_1, z_3), (z_1, z_4) und (z_1, z_6) beschränken, wenn beispielsweise die Aktivitäten *Releaseplanung (AE)* und *Versionsmanagement (AE)* das Anforderungsdokument (über die gleichnamige Schnittstelle) nicht direkt von der Aktivität *Anforderungsanalyse erstellen* erhalten.

Die in Abbildung 3.8 verwendete Dokumentensicht ist auch für die Identifikation von Kommunikationskanälen zwischen verschiedenen Orten hilfreich. Abbildung 3.9 zeigt dies an einem Beispiel, in dem das Qualitätsmanagement Richtlinien – u.a. zur Programmierung – an das Component Engineering schickt.

In dieser verteilten Prozesslandschaft zur komponentenbasierten Softwareentwicklung existieren zwei Standorte, die die Aktivitäten des Qualitätsmanagements enthalten, sowie insgesamt fünf Standorte, an denen Komponenten entwickelt werden. Es gilt:

$$local(\text{Qualitätsmanagement}) = \{o_1, o_2\}$$
$$local(\text{Component Engineering}) = \{o_1, o_2, o_3, o_4, o_5\}$$

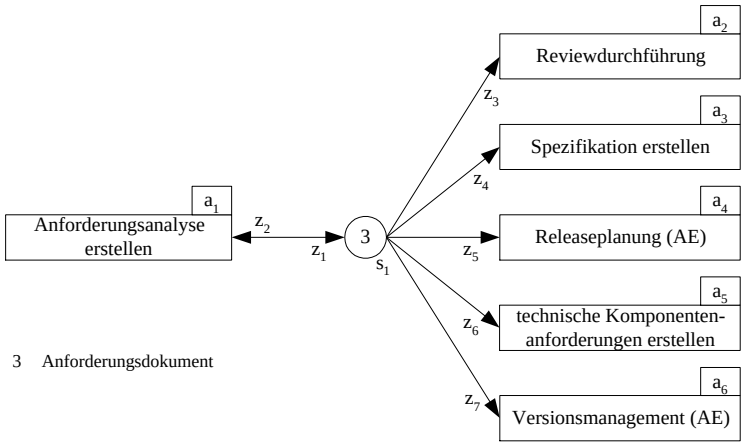


Abbildung 3.8: Kommunikationskanäle in einer Prozesslandschaft

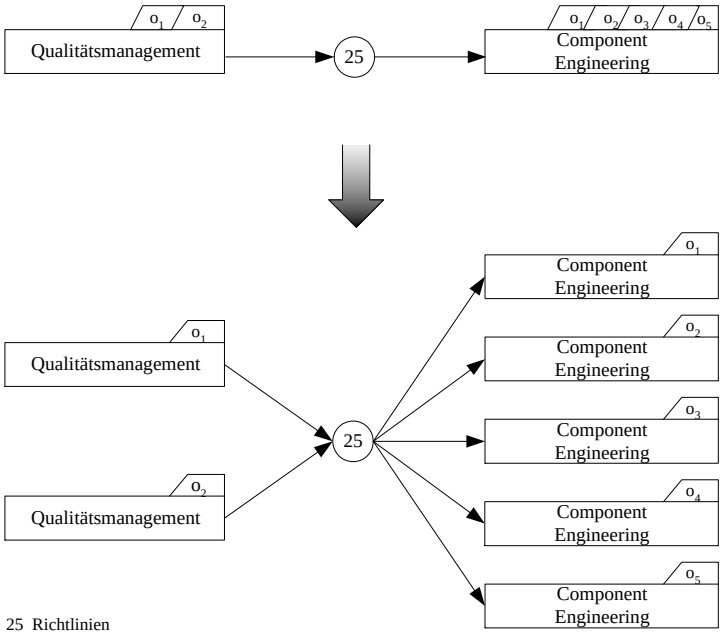


Abbildung 3.9: Kommunikationskanäle in einer verteilten Prozesslandschaft

Damit werden an den Orten o_1 und o_2 jeweils Aktivitäten des Qualitätsmanagements und des Component Engineering durchgeführt. An den Orten o_3 bis o_5 finden sich dagegen ausschließlich Aktivitäten des Component Engineering. Wird nun die Dokumentensicht auf das Richtliniendokument derart erweitert, dass jede Aktivität in der Häufigkeit ihres Vorkommens abgebildet wird und auf die Richtlinien zugreift, resultiert daraus der untere Teil der Abbildung 3.9. Hier kann ein Modellierer beispielsweise festlegen, dass das Qualitätsmanagement am Ort o_1 ausschließlich mit dem Component Engineering am selben Ort kommuniziert, während das Qualitätsmanagement am Ort o_2 sich um alle weiteren Komponentenentwicklungsstandorte kümmert. Aus der Menge aller theoretisch möglichen Kommunikationskanäle werden so nur diejenigen ausgewählt, die tatsächlich vorkommen.

Definition 3.1.38. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft, PL_{top} vollständig, $CC \neq \emptyset$ eine Menge von Kommunikationskanälen und*

$$K := switch^6$$

das sechsfache kartesische Produkt der Menge $switch$, wobei jedes $k \in K$ ein geordnetes Tupel ist. Mit der Abbildung $\mathbf{value_Z}$ wird jedem Zugriff $z \in Z$ eine konkrete Ausprägungskombination der Kommunikationsattribute zugeordnet:

$$value_Z : Z \rightarrow K \qquad mit value_Z(z) = (k_1, ..., k_6)$$

Die Abbildung gibt mit

k_1 *die Ausprägung der Persistenz,*
k_2 *die Ausprägung der Synchronität,*
k_3 *die Ausprägung der Veränderbarkeit,*
k_4 *die Ausprägung der Kodierung,*
k_5 *die Ausprägung der Privatheit und*
k_6 *die Ausprägung des Mehrfachversands an.*

Bemerkung 3.1.39. *Für einen funktionsfähigen Kommunikationskanal müssen bis auf die Persistenz die Attributwerte derjenigen Kommunikationsattribute zusammenpassen, die den beteiligten Zugriffen zugeordnet sind. Dies ist immer dann der Fall, wenn die Attributwerte eines Kommunikationskanals jeweils pro Attribut identisch sind. Die Persistenz wird für die Funktionsfähigkeit eines Kommunikationskanals nicht betrachtet.*

Die bislang in diesem Abschnitt definierten Abbildungen zur Unterstützung der statischen Analyse einer Prozesslandschaft führen zu ihrer nächsthöheren Ausbaustufe:

Definition 3.1.40. *Sei $PL_{top} = (A, S, Z, AB)$ die erste Ausbaustufe einer Prozesslandschaft, PL_{top} vollständig,*

- $switch$ *die Menge aller möglichen Attributwerte von Zugriffen,*

- $ZZ \subset Z \times Z$,
- $CC \neq \emptyset$ *eine Menge von funktionsfähigen Kommunikationskanälen,*
- K *das sechsfache kartesische Produkt der Menge* $switch$ *wie in Definition 3.1.38 festgelegt,*
- per, $synch$, $change$, $coded$, $priv$ *und* $mult$ *die in Definition 3.1.35 eingeführten Abbildungen zur Festlegung der Kommunikationsattribute,*
- $channel$ *eine Abbildung zur Festlegung von Kommunikationskanälen gemäß Definition 3.1.37 und schließlich*
- $value_Z$ *die Abbildung aus der vorangegangenen Definition 3.1.38.*

Damit lässt sich die ***zweite Ausbaustufe einer Prozesslandschaft*** *definieren als ein Tupel*

$PL_{top-com} :=$
$(PL_{top}, switch, ZZ, CC, K, per, synch, change, coded, priv, mult, channel, value_Z)$

Notation 3.1.41. *Wenn keine Missverständnisse auftreten können, wird eine Prozesslandschaft* $PL_{top-com}$ = $(PL_{top}, switch, ZZ, CC, K, per, synch, change, coded, priv, mult, channel, value_Z)$ *im Folgenden verkürzt als* $PL_{top-com} = (A, S, Z, AB)$ *bezeichnet.*

Im Folgenden werden grundsätzlich funktionsfähige Kommunikationskanäle betrachtet. Dazu wird zunächst die Abbildung $value_Z$ erweitert als Abbildung von der Menge der Kommunikationskanäle auf das kartesische Produkt K. Dies erleichtert eine Zuweisung von konkreten Attributausprägungen an die beteiligten Zugriffe, da so statt zwölf nur noch sechs Werte pro Kommunikationskanal zuzuweisen sind.

Definition 3.1.42. *Sei* $PL_{top-com} = (A, S, Z, AB)$ *die zweite Ausbaustufe einer Prozesslandschaft und* $CC \neq \emptyset$ *die zugehörige Menge funktionsfähiger Kommunikationskanäle. Sei weiterhin K das sechsfache kartesische Produkt der Menge* $switch$ *wie in Definition 3.1.38 festgelegt. Mit der Abbildung* $\mathbf{value_{CC}}$ *wird jedem Kommunikationskanal* $cc \in CC$ *mit* $cc = (z_1, z_2), z_1, z_2 \in Z$ *eine konkrete Ausprägungskombination der Kommunikationsattribute zugeordnet:*

$$value_{CC} : CC \rightarrow K \text{ mit}$$

$$value_{CC}(z_1, z_2) := value_Z(z_1)$$

Für funktionsfähige Kommunikationskanäle wird nachfolgend eine Begriffssammlung zur Analyse von *Komplexität und Dichte der Kommunikationsinfrastruktur einer Prozesslandschaft* definiert. Die Dichte einer Kommunikationsinfrastruktur ist abhängig von der Anzahl vorhandener Kommunikationskanäle innerhalb einer betrachteten Prozesslandschaft bzw. innerhalb einer betrachteten Teilmenge. Die Komplexität der Kommunikationsinfrastruktur einer Prozesslandschaft wird beeinflusst von den Attributen

- Persistenz (Aufwand für Datenhaltung),
- Synchronität (technisch unterschiedliche Kommunikationsvoraussetzungen),
- Veränderbarkeit (Verfügbarkeit von z.B. Signaturmechanismen),
- Verschlüsselung (Verfügbarkeit von Kodierungs- und Sicherheitsmechanismen),
- Privatheit (Aufwand für Zugriffs- und Schlüsselverwaltung) und
- Mehrfachversand (nicht immer standardmäßig verfügbar).

Je unterschiedlicher die Attribute für eine Menge von Kommunikationskanälen ausgeprägt sind und je vielfältiger letztere damit sind, desto komplexer ist die korrespondierende Infrastruktur.

Definition 3.1.43. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle. Mit Hilfe der Abbildung $value_{CC}$ lässt sich die **Komplexität** einer Kommunikationsinfrastruktur definieren als:*

$$Komplexität := |\{value_{CC}(cc) \mid \exists\, z_1, z_2 \in Z : channel(z_1, z_2) = cc\}|$$

Kopplung von Orten

Unter dem Oberbegriff der Kopplung von Orten wird die Kommunikation zwischen Aktivitäten an verschiedenen ausgewählten Orten betrachtet. Der Begriff der Kopplung ist dabei, wie bereits zu Beginn des Abschnitts 2.3.4 motiviert, dem Bereich der komponentenbasierten Softwareentwicklung entliehen.

Definition 3.1.44. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle. Weiterhin seien $o_1, o_2 \in O$ mit $o_1 \neq o_2$, $s \in S$,*

$$\begin{aligned} A(o) :=& \{a \in leaves \mid local(a) = \{o\}\} \text{ und} \\ CC(o) :=& \{cc \in CC \mid \exists\, a \in A(o)\, \exists\, a' \notin A(o) : \\ & cc = channel\big((a, s), (s, a')\big) \vee cc = channel\big((a', s), (s, a)\big)\} \end{aligned}$$

diejenige Teilmenge aller Kanäle $cc \in CC$, über die jeweils eine Aktivität $a \in A(o)$ mit einer Aktivität $a' \notin A(o)$ kommuniziert.

*Die **Kopplungsdichte zwischen zwei Orten** o_1 und o_2 entspricht der Anzahl der Kommunikationskanäle zwischen allen Paaren von Aktivitäten a und d der beiden ausgewählten Orte o_1 und o_2:*

$$Kopplungsdichte_2(o_1, o_2) := |CC(o_1) \cap CC(o_2)|$$

Bemerkung 3.1.45. *Der Index der Kopplungsdichte zeigt die Anzahl der Orte an, für die diese Kopplungsdichte berechnet wird.*

Beispiel:

Im Beispiel der Prozesslandschaft zur komponentenbasierten Softwareentwicklung sind die meisten der Projektmanagementaktivitäten einem Standort o_1 = $Managementstandort$ zugeordnet. An diesem finden alle koordinierenden Aktivitäten wie Projektplanung, Ressourcenverteilung und Projektsteuerung statt. Außerdem werden hier wichtige Entscheidungen wie Einkauf oder Neuentwicklung benötigter Komponenten getroffen. An einem zweiten Standort o_2 = $Anwendungsentwicklung$ wird die Anwendung für den Kunden entwickelt. Dazu werden Spezifikationen erstellt, die Softwarearchitektur entworfen und schließlich die gekauften und die durch das Component Engineering realisierten Komponenten zu einem Gesamtsystem zusammengesetzt. Außerdem werden an diesem Ort alle begleitenden Reviews und Tests des zu erstellenden Systems durchgeführt.

Tabelle 3.2 zeigt die Kommunikation zwischen den beiden Standorten. Jede Zeile repräsentiert einen Kommunikationskanal. Insgesamt werden elf Dokumente vom Managementstandort zur Anwendungsentwicklung und 13 Dokumente von der Anwendungsentwicklung zum Managementstandort geschickt. Daraus ergibt sich für die Kopplungsdichte ein Wert von 24 bei 20 Aktivitäten am Managementstandort und 16 Aktivitäten bei der Anwendungsentwicklung.

Definition 3.1.46. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $o \in O$. Die **Kopplungsdichte eines Ortes** o ist definiert als*

$$Kopplungsdichte_1(o) := \sum_{o' \in O \setminus \{o\}} Kopplungsdichte_2(o, o')$$

Die Kopplungsdichte eines Ortes entspricht damit der Anzahl der Kommunikationskanäle zwischen allen Paaren von Aktivitäten $a \in A(o)$ und $d \notin A(o)$, bei denen also $local(a) = \{o\}$ und $\{o\} \notin local(a')$. Sie wird durch die Anzahl seiner ein- und ausgehenden Kommunikationskanäle bestimmt.

Beispiel:

Abbildung 3.10 zeigt einen Überblick über die Kopplungsdichten der Beispielprozesslandschaft, die hier über insgesamt vier Standorte verteilt ist. Neben dem Managementstandort und der Anwendungsentwicklung gibt es noch den Kundenstandort und die Komponentenentwicklung. Am Standort des Kunden werden beispielsweise das Anforderungsdokument erstellt und die Abnahme durchgeführt, während bei der Komponentenentwicklung der Entwurf und die Implementierung der neu zu entwickelnden Komponenten durchgeführt werden. Die Zahlen an den Verbindungslinien zwischen den einzelnen Standorten entsprechen der Kopplungsdichte. Die Kopplungsdichte des Managementstandortes (= 38) lässt sich für dieses Beispiel als Summe der Kopplungsdichten zum Kundenstandort, zur Anwendungs- und zur Komponentenentwicklung berechnen.

Sender-Aktivität	**Ort**	**Empfänger-Aktivität**	**Ort**	**Dokument**
Releaseplanung	o_2	Projektplanerstellung	o_1	Releaseplanung
Buy-or-Build Entscheidung treffen	o_1	Komponente kaufen	o_2	Buy-or-Build Entscheidung
Projekt einschätzen	o_1	Releaseplanung	o_2	Projekteinschätzung
Wiederverwendungs-repository verwalten	o_1	Spezifikation erstellen	o_2	Repository Spezifikation
Wiederverwendungs-repository verwalten	o_1	Systemarchitektur erstellen	o_2	Repository Architektur
Wiederverwendungs-repository verwalten	o_1	Vorh. Komponenten auswählen	o_2	Repository Softwarekomponenten
Geschäftsprozess modellieren	o_1	Spezifikation erstellen	o_2	Repository Spezifikation
Testdurchführung	o_2	Testablauf-verfolgung	o_1	Status Test
Testvorbereitung	o_2	Testauswertung	o_1	Testbeschreibung
Testdurchführung	o_2	Testauswertung	o_1	Fehlerprotokoll integr. Softwaresystem
Testdurchführung	o_2	Testauswertung	o_1	Fehlerprotokoll install. Softwaresystem
Testdurchführung	o_2	Testauswertung	o_1	Fehlerprotokoll Neue Komponente
Richtlinienerstellung	o_1	Testvorbereitung	o_2	Richtlinien
Richtlinienerstellung	o_1	Testdurchführung	o_2	Richtlinien
Richtlinienerstellung	o_1	Reviewdurchführung	o_2	Richtlinien
Testplanerstellung	o_1	Testvorbereitung	o_2	Testplan
Testablaufverfolgung	o_1	Testvorbereitung	o_2	Testplan
Reviewdurchführung	o_2	Testablaufverfolgung	o_1	Status Review
Reviewdurchführung	o_2	Reviewauswertung	o_1	Review Anforderungsdokument
Reviewdurchführung	o_2	Reviewauswertung	o_1	Review Spezifikationsdokument
Reviewdurchführung	o_2	Reviewauswertung	o_1	Review Systemarchitektur
Reviewdurchführung	o_2	Reviewauswertung	o_1	Review Releaseplanung
Reviewdurchführung	o_2	Reviewauswertung	o_1	Review Technische Komponentenanforderung
Komponente anfordern	o_2	Markt analysieren	o_1	Komponentenanforderung

Tabelle 3.2: Kommunikation zwischen dem Managementstandort und dem Standort *Anwendungsentwicklung*

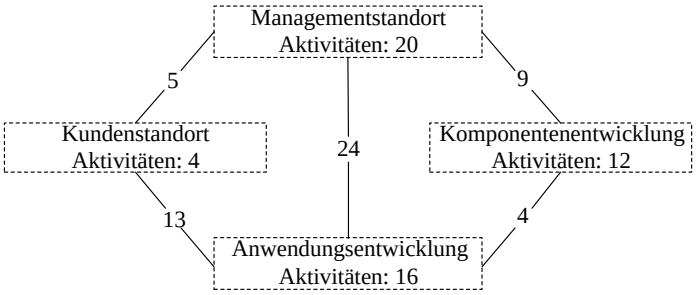


Abbildung 3.10: Berechnung der Kopplungsdichte eines Ortes

Über die **Kopplungskomplexität** von Orten lässt sich die Komplexität der Kommunikationsstruktur einer Prozesslandschaft bestimmen.

Definition 3.1.47. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft, $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle und $o_1, o_2 \in O$ mit $o_1 \neq o_2$. Weiterhin sei mit $CC(o)$ eine Teilmenge der funktionsfähigen Kommunikationskanäle wie in Definition 3.1.44 festgelegt. Die **Kopplungskomplexität zwischen zwei Orten** o_1 und o_2 bezeichnet die Anzahl der verschiedenartig ausgeprägten Kommunikationskanäle zwischen allen Paaren von Aktivitäten $a, a' \in A$ der beiden ausgewählten Orte o_1 und o_2:*

$$Kopplungskomplexität_2(o_1, o_2) := \left|\{value_{CC}(cc) \mid cc \in CC(o_1) \cap CC(o_2)\}\right|$$

Definition 3.1.48. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $o \in O$. Die **Kopplungskomplexität eines Ortes** o ist definiert als*

$$Kopplungskomplexität_1(o) := \sum_{o' \in O \setminus \{o\}} Kopplungskomplexität_2(o, o')$$

Die Kopplungskomplexität eines Ortes bezeichnet damit die Anzahl der verschiedenartig ausgeprägten Kommunikationskanäle zwischen allen Paaren von Aktivitäten $a \in A(o)$ und $a' \notin A(o)$, bei denen also $local(a) = \{o\}$ und $\{o\} \notin local(a')$ ist.

Beispiele:

- *Fall 1:* Innerhalb einer Prozesslandschaft ist geplant, Standorte zusammenzulegen. Eine hohe Kopplungskomplexität zwischen zwei Orten deutet darauf hin, dass vorzugsweise diese zwei Orte zusammengelegt werden sollten, um den Aufwand für den Aufbau einer Kommunikationsinfrastruktur zu minimieren. Für die Identifikation der maximalen Kopplungskomplexität einer Prozesslandschaft wird diese für alle möglichen Kombinationen von jeweils zwei Orten berechnet.

- *Fall 2:* Innerhalb einer Prozesslandschaft werden die Auflösung eines oder mehrerer Standorte und die Verteilung der dortigen Aktivitäten auf andere Standorte geplant. Es empfiehlt sich, eher Aktivitäten von Orten mit geringer Kopplungskomplexität auf mehrere Standorte zu verteilen statt Aktivitäten von Orten mit hoher Kopplungskomplexität, um die Kostensteigerung der verbleibenden Standorte möglichst gering zu halten. Dazu wird das Minimum der Kopplungskomplexitäten aller Orte berechnet.
- *Fall 3:* Ergibt die Analyse einer gegebenen Prozesslandschaft eine hohe Kopplungskomplexität zu externen Aktivitäten (z.B. Zulieferer), sollte individuell überlegt werden, ob dies erwünscht ist. So kann es z.B. sein, dass eine hohe Kopplung an Zuliefereraktivitäten eine große Abhängigkeit anzeigt, die immer vermieden werden sollte.

Die **Kopplungszahl** eines Ortes wird ähnlich der Kopplungsdichte eines Ortes definiert, allerdings wird hierfür lediglich die Anzahl derjenigen Orte gezählt, die über Kommunikationskanäle mit dem betrachteten Ort verbunden sind.

Definition 3.1.49. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $o \in O$. Die **Kopplungszahl** eines Ortes o entspricht der Anzahl der Orte, die über Kommunikationskanäle zwischen allen Paaren von Aktivitäten a und d verbunden sind, bei denen $local(a) = \{o\}$ und $o \notin local(d')$ gilt:*

$$Kopplungszahl(o) := \big|\{o' \in O \backslash \{o\} \mid Kopplungsdichte_2(o, o') \geq 1\}\big|$$

Die Kopplungszahl eines Ortes bezeichnet also die Anzahl der Orte, mit denen ein ausgewählter Ort kommuniziert. Der Managementstandort der Prozesslandschaft in Abbildung 3.10 hat beispielsweise eine Kopplungszahl von drei.

Kohäsion von Orten

Kohäsion in Prozesslandschaften ist ähnlich aufzufassen wie Kohäsion im Rahmen komponentenbasierter Softwareentwicklung. Es werden Anzahl und Vielfalt der Kommunikationskanäle zwischen Aktivitäten eines Ortes betrachtet.

Definition 3.1.50. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle. Weiterhin seien $s \in S$, $o \in O$ und $A(o) := \{a \in leaves \mid local(a) = \{o\}\}$. Schließlich sei*

$$CC'(o) := \{cc \in CC \mid \exists\, a, a' \in A(o) : cc = channel\big((a, s), (s, a')\big) \vee cc = channel\big((a', s), (s, a)\big)\}$$

diejenige Teilmenge aller Kommunikationskanäle $cc \in CC$, über die jeweils zwei am gleichen Ort o liegende Aktivitäten $a, d' \in A$ miteinander kommunizieren.

*Die **Kohäsionsdichte** eines Ortes o entspricht der Anzahl der funktionsfähigen Kommunikationskanäle zwischen allen Paaren von Aktivitäten a, d an diesem Ort o:*

$$Kohäsionsdichte(o) := \big|CC'(o)\big|$$

Tabelle 3.3 zeigt die interne Kommunikation am Managementstandort. Jede Zeile repräsentiert einen Kommunikationskanal, woraus sich insgesamt eine Kohäsionsdichte von 20 ergibt.

Sender-Aktivität	**Dokument**	**Empfänger-Aktivität**
Ressourcenplanung	Ausstattung	Projekt initialisieren
Teamzusammenstellung	Projektteam	Projekt initialisieren
Projektplanerstellung	Projektplan	Projekt initialisieren
Risikomanagement	Abnahmefreigabe	Projektsteuerung und -kontrolle
Projektsteuerung und -kontrolle	Projektstatus	Projektdarstellung nach außen
Markt analysieren	Marktanalyse	Buy or Build-Entscheidung treffen
Buy or Build-Entscheidung treffen	Buy or Build-Entscheidung	CE mit Komponenten-entwicklung beauftragen
Projekt einschätzen	Projekteinschätzung	Risikomanagement
Projekt einschätzen	Projekteinschätzung	Markt analysieren
Projekt einschätzen	Projekteinschätzung	Projektsteuerung und -kontrolle
Geschäftsprozess modellieren	Geschäftsprozessmodell	Wiederverwendungs-repository verwalten
Wiederverwendungsrepository verwalten	Repository Architektur	Projekt einschätzen
Wiederverwendungsrepository verwalten	Repository Softwarekomponenten	Projekt einschätzen
Richtlinienerstellung	Richtlinien	Reviewauswertung
Richtlinienerstellung	Richtlinien	Testauswertung
Testauswertung	Testauswertung	Richtlinienerstellung
Testablaufverfolgung	Testplan	Reviewauswertung
Testplanerstellung	Testplan	Testablaufverfolgung
Testplanerstellung	Testplan	Projektplanerstellung
Reviewauswertung	Reviewauswertung	Richtlinienerstellung

Tabelle 3.3: Interne Kommunikation am Managementstandort

Definition 3.1.51. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft, $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle und $o \in O$. Sei weiterhin mit $CC'(o)$ eine Teilmenge der funktionsfähigen Kommunikationskanäle wie in Definition 3.1.50 festgelegt.*

*Die **Kohäsionskomplexität** eines Ortes o entspricht der Anzahl unterschiedlich ausgeprägter Kommunikationskanäle zwischen allen Paaren von Aktivitäten a, $\acute{d}$ an diesem Ort o:*

$$Kohäsionskomplexität(o) := \left|\{value_{CC}(cc) \mid cc \in CC'(o)\}\right|$$

Für das Beispiel des Managementstandortes wurde eine Kohäsionskomplexität von 12 berechnet. Die Kommunikation verläuft mehrheitlich asynchron, da Informationen über Dokumente ausgetauscht werden, deren Inhalt lediglich bei Besprechungsterminen synchron vermittelt wird. Kombinationen von Attributwerten, die in diesem Beispiel nicht vorkommen, sind unter anderem ein verschlüsselter Mehrfachversand, veränderbarer und synchroner Informationsaustausch oder eine gleichzeitig asynchrone und verschlüsselte Kommunikation.

Definition 3.1.52. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft, $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle, $s \in S$ und $o \in O$. Weiterhin sei mit $A(o)$ eine Aktivitätsteilmenge wie in Definition 3.1.50 festgelegt.*

*Die **Kohäsionszahl** entspricht der Anzahl der Aktivitäten an einem Ort o, die über Kommunikationskanäle mit anderen Aktivitäten am selben Ort o verbunden sind:*

$$Kohäsionszahl(o) := \big|\{a \in A(o) \mid \exists\, a' \in A(o) : cc = channel\big((a,s),(s,a')\big) \vee cc = channel\big((a',s),(s,a)\big)\}\big|$$

Die Kohäsionszahl des Managementstandortes der Beispielprozesslandschaft beträgt 19. Die Differenz zur Gesamtzahl von 20 Aktivitäten (vgl. Abbildung 3.10) erklärt sich dadurch, dass die Aktivität *Projektabschluss* zwar externe Kommunikation mit dem Kunden betreibt, intern aber – zumindest bei dem modellierten Abstraktionsgrad – keine Informationen weitergeleitet werden.

Mit den Definitionen rund um die Begriffe der Kohäsion und Kopplung von Aktivitäten ist ein Rahmenwerk entstanden, das die Analyse der Kommunikationsinfrastruktur der zweiten Ausbaustufe einer gegebenen Prozesslandschaft erlaubt. Wie die konkreten Kopplungs- und Kohäsionswerte zur Berechung und Analyse eines Kommunikationskostenfaktors Kkf eingesetzt werden, ist bereits in Abschnitt 2.3.4 auf Seite 44 erläutert worden. Die Analyse bleibt auf statischer Ebene, da die verwendete PLL-Notation keinerlei Information über die Dynamik der Kommunikation enthält. Um für eine gegebene Prozesslandschaft, die in PLL-Notation dargestellt ist, ihr dynamisches Verhalten betrachten zu können, muss diese zunächst um Ablaufinformationen ergänzt werden. Diese Ergänzungen sowie zusätzliche Erweiterungen des zugrundeliegenden Petrinetzes sind Thema des nachfolgenden Abschnitts.

3.2 Notation der unteren Ebenen einer Prozesslandschaft

Die unteren Ebenen einer Prozesslandschaft unterscheiden sich insofern von den oberen, als dass auf den unteren immer auch Ablaufinformationen modelliert sind. Die Ergänzung der zweiten Ausbaustufe einer Prozesslandschaft um diese Ablaufinformationen und damit der Übergang von der Modellierung der oberen zur Modellierung der unteren Ebenen wird in Abschnitt 3.2.1 diskutiert. In Abschnitt 3.2.2 werden weitere Abbildungen und Begriffe formuliert, die für eine vierte Ausbaustufe einer Prozess-

landschaft die Analyse der unteren Landschaftsebenen bezüglich verschiedener dynamischer Kommunikationseigenschaften ermöglichen.

3.2.1 Hinzufügen von Ablaufinformationen

Die Ergänzung einer in PLL notierten Prozesslandschaft um Ablaufinformationen erfordert die Berücksichtigung verschiedener Aspekte. Zunächst wird das zugrundeliegende Petrinetz durch das Hinzufügen weiterer Zugriffe $z \in Z$ zu einem zusammenhängenden Graphen ergänzt. Für diesen werden Bedingungen formuliert, wann eine Aktivität schaltfähig ist, um darüber vom Kontext der betrachteten Prozesslandschaft abhängige Verhaltensvorschriften festlegen zu können. Die einzelnen hierfür notwendigen Schritte werden in der Reihenfolge ihrer Durchführung diskutiert. Zuvor werden jedoch einige grundlegende Begriffe definiert, die bestimmte Eigenschaften von Petrinetzen charakterisieren. Sie werden für die schrittweise Ergänzung benötigt.

Definition 3.2.1. *Sei $R \subseteq X \times X$ eine binäre Relation und $x, y \in X$.*

$$\bullet x := \{y \mid yRx\} \text{ heißt der } \textbf{\textit{Vorbereich}} \text{ von x.}$$
$$x\bullet := \{y \mid xRy\} \text{ heißt der } \textbf{\textit{Nachbereich}} \text{ von x.}$$

Des Weiteren wird definiert:

$$\bullet X := \bigcup_{x \in X} \bullet x \text{ und } X\bullet := \bigcup_{x \in X} x\bullet .$$

Insbesondere gilt also für $x, y \in X$: $x \in \bullet y \Leftrightarrow y \in x\bullet$.

Definition 3.2.2. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein Petrinetz. PN heißt **proper**:$\Leftrightarrow$*

1. $S \cup A_{PN} \neq \emptyset$
2. $\bullet(S \cup A_{PN}) \cup (S \cup A_{PN})\bullet = S \cup A_{PN}$

Bedingung 1. bedeutet, dass ein Netz ohne Knoten ausgeschlossen ist. Bedingung 2. bedeutet, dass es keine isolierten Knoten gibt. Dadurch wird ein minimaler lokaler Zusammenhang gefordert.

Definition 3.2.3. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein properes Petrinetz. Seien weiter $k, k' \in A_{PN} \cup S$. Eine Folge $z_1, ..., z_n$ von Zugriffen heißt **ungerichteter Pfad** von k nach k' :$\Leftrightarrow$ Es existiert eine Folge*

$$k = k_1, k_2, ..., k_n, k_{n+1} = k' \qquad \text{von } k \text{ nach } k'$$

von Knoten aus $A_{PN} \cup S$, so dass für alle $1 \leq i \leq n$ gilt:

$$z_i = (k_i, k_{i+1}) \in Z_{PN} \vee z_i = (k_{i+1}, k_i) \in Z_{PN}$$

*PN heißt **kohärent** :$\Leftrightarrow$ Für alle $k, k' \in (A_{PN} \cup S)$ existiert ein ungerichteter Pfad von k nach k'.*

Durch die Kohärenz wird ein vollständiger Zusammenhang zwischen den Knoten aus A_{PN} und S gefordert.

Um für eine über mehrere Orte verteilte Prozesslandschaft deren Kommunikationsverhalten analysieren zu können, werden für die Struktur des zugrundeliegenden Petrinetzes Einschränkungen formuliert, die „Synchronieaussagen“ [Bau96] ermöglichen. Diese beantworten unter anderem „Fragen der Art, wie oft Ereignisse einer Art A und Ereignisse einer Art B im Verhältnis zueinander stattfinden können“ [Bau96]. Bezogen auf die Analyse von Kommunikationsverhalten einer verteilten Prozesslandschaft können dies Fragen nach den internen Kommunikationskosten im Verhältnis zu externen Kommunikationskosten sein oder Fragen zum Verhältnis der Prozessauslastungen verschiedener Teilbereiche einer Prozesslandschaft zueinander (vgl. Abschnitt 3.2.2.2). Für das Beispiel der komponentenbasierten Softwareentwicklung muss dazu unter anderem messbar sein, wie oft das Qualitätsmanagement Reviews für welche Komponentenentwicklungsstandorte durchzuführen hat. Dazu muss auch eindeutig erkennbar sein, welcher Standort ein Dokument zum Review an das Qualitätsmanagement schickt. Letzteres ist aber nur dann möglich, wenn im zugrundeliegenden Petrinetz jede Aktivität eines Komponentenentwicklungsstandortes über einen eigenen Kommunikationskanal – und damit über eine separate Schnittstelle – mit einer Aktivität des Qualitätsmanagements kommuniziert. Würde die Kommunikation mehrerer Komponentenentwicklungsstandorte mit einem Qualitätsmanagementstandort über eine gemeinsame Schnittstelle modelliert, wäre die eindeutige Identifikation der Herkunft eines weitergeleiteten Informationsobjektes nicht mehr klar nachvollziehbar. Für eine Prozesslandschaft werden dazu entsprechende Einschränkungen für die Menge der Schnittstellen formuliert.

Definition 3.2.4. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft. Für die Menge S der Schnittstellen können zwei ausgezeichnete Teilmengen definiert werden.*

$$S_{extern} := \{s \in S \big| \, |\bullet s| = |s \bullet| = 1 \ \wedge \ \bullet s \cap s\bullet = \emptyset\}$$
$$S_{intern} := S \backslash S_{extern}$$

Bemerkung 3.2.5. *Durch S_{intern} und S_{extern} wird die Menge S der Schnittstellen in zwei disjunkte Teilmengen unterteilt, d.h.*

$$S = S_{intern} \cup S_{extern} \text{ mit } S_{intern} \cap S_{extern} = \emptyset$$

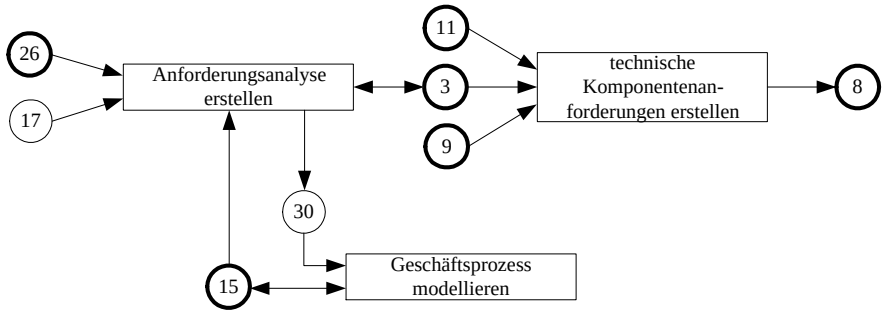
Notation 3.2.6. *S_{intern} bezeichnet eine Menge von **internen Schnittstellen**, S_{extern} eine Menge von **externen Schnittstellen**.*

Bemerkung 3.2.7. *Eine externe Schnittstelle gibt an, dass es eine Verbindung zwischen zwei Aktivitäten auf oberen Abstraktionsniveaus gibt und damit mindestens eine Schnittstelle zwischen zwei Aktivitäten innerhalb von verfeinerten bzw. noch zu verfeinernden Aktivitäten. Die zugehörigen Einschränkungen sind erforderlich um sicherzustellen, dass die Grenzen zwischen zwei verfeinerten Aktivitäten einfach handhabbare*

Schnittstellen bleiben, an denen keine Konflikte auftreten. Letztere sind immer dann gegeben, wenn zwei Aktivitäten mindestens eine gemeinsame (markierte) Schnittstelle in ihren Vorbereichen haben.

Alle für das Hinzufügen von Ablaufinformationen zu einer als $PL_{top-com}$ vorliegenden Prozesslandschaft erforderlichen Schritte werden nach der Einführung notwendiger Definitionen und Abbildungsvorschriften jeweils beispielhaft an Ausschnitten der Prozesslandschaft zur komponentenbasierten Softwareentwicklung erläutert, die in Abschnitt 2.2.1 eingeführt worden ist. Diese Schritte beinhalten

1. die Erweiterung der Prozesslandschaft $PL_{top-com}$ zu einer kohärenten Prozesslandschaft,
2. das Festlegen der Schaltverhalten für alle Aktivitäten innerhalb der Prozesslandschaft,
3. die Erweiterung der Verfeinerungskonzepte bezüglich Aktivitäten und Schnittstellen und
4. die Berücksichtigung von speziellen Situationen innerhalb der Prozesslandschaft.



3 Anforderungsdokument
8 technische Komponentenanforderung
9 Review technische Komponentenanforderung
11 Systemarchitektur
15 Geschäftsprozessmodell
17 erste Anforderungen
26 Review Anforderungsdokument
30 Anfrage nach Geschäftsprozessmodell

Abbildung 3.11: Teilausschnitt der Beispielprozesslandschaft

Abbildung 3.11 zeigt einen Ausschnitt der in PLL modellierten Prozesslandschaft zur komponentenbasierten Softwareentwicklung. Für die Diskussion der Schritte 1. bis 4. werden Aktivitäten des Application Engineering und des Domain Engineering näher betrachtet, die auf die acht in der Abbildung dargestellten Schnittstellen zugreifen. Die

beiden Schnittstellen *erste Anforderungen* und *Anfrage nach Geschäftsprozessmodell* werden nur innerhalb des dargestellten Teilbereichs verwendet und sind daher nicht fett umrandet. Der in Abbildung 3.11 dargestellte Teilbereich der Prozesslandschaft ist für die Diskussion repräsentativ, da sowohl verschiedene Abstraktionsebenen und Aktivitäten aus verschiedenen Verfeinerungen als auch unterschiedliche Zugriffe und Schnittstellen betrachtet werden. Die Landschaftselemente (Aktivitäten, Schnittstellen, Zugriffe) sind in Abbildung 3.11 als eigene Prozesslandschaft dargestellt, um den Kontext der Aktivitäten und ihrer Zugriffe auf die Schnittstellen zu verdeutlichen. Im korrespondierenden Aktivitätenbaum stehen sie in keiner Hierarchierelation zueinander.

Bevor eine Prozesslandschaft $PL_{top-com}$ zu einer kohärenten Landschaft erweitert werden kann, muss sie verschiedene Voraussetzungen erfüllen. Ihre Vollständigkeit (vgl. Definition 3.1.32, Seite 81) ist eine Mindestvoraussetzung. Sie gewährleistet die Erstellung einer kohärenten Prozesslandschaft, die nicht aus isolierten Fragmenten besteht. Für eine spätere Darstellung von Ablaufinformationen sollte ein Modellierer außerdem zunächst überprüfen, ob jede Aktivität über einen lesenden Zugriff mit einer Schnittstelle verknüpft ist. Existiert kein solcher Zugriff bzw. keine korrespondierende Schnittstelle, entsteht in der kohärenten Prozesslandschaft eine Aktivität ohne Vorbereich, die somit immer schaltfähig ist (vgl. Definition 3.2.15, Seite 104). Dies ist aber aus Ablaufsicht selten so gewollt, und daher sollte ggf. eine entsprechende Schnittstelle bereits in der PLL-Notation ergänzt werden. Eine Ergänzung dieser Schnittstelle zu einem späteren Zeitpunkt hätte den Nachteil, dass nachträglich nicht nur weitere Datenflussrelationen modelliert, sondern auch bereits definierte Schaltregeln (vgl. Abschnitt 3.2.1.2) erneut überprüft und eventuell angepasst werden müssten.

Im Beispielausschnitt der Prozesslandschaft in Abbildung 3.11 ist die Schnittstelle *Anfrage nach Geschäftsprozessmodell* mit einem lesenden Zugriff seitens der Aktivität *Geschäftsprozess modellieren* aus dem zuletzt angesprochenen Grund ergänzt worden. In der erweiterten Prozesslandschaft wird sie zum Vorbereich der Aktivität *Geschäftsprozess modellieren*, ohne dessen Belegung diese nicht schalten kann.

3.2.1.1 Erweiterung einer Prozesslandschaft zu einem kohärenten Netz

Die Erweiterung einer Prozesslandschaft zu einem kohärenten Netz berücksichtigt zunächst nur nicht weiter verfeinerte Aktivitäten zusammen mit ihren expliziten Zugriffen auf Schnittstellen. Existieren in einer Prozesslandschaft $PL_{top-com}$ mehrere Zugriffe von Aktivitäten auf eine Schnittstelle s, muss diese für die Konstruktion einer kohärenten Prozesslandschaft mehrfach dupliziert werden. Die Anzahl der resultierenden Schnittstellen ergibt sich aus der Anzahl der auf s schreibend zugreifenden Aktivitäten multipliziert mit der Anzahl der auf s lesend zugreifenden Aktivitäten, sofern nicht ausschließlich lesende oder ausschließlich schreibende Zugriffe auf die Schnittstelle s existieren. So wird sichergestellt, dass zunächst alle möglichen Datenflüsse abgebildet sind.

Definition 3.2.8. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $S = \{s_1, ..., s_n\}$.*
Eine Prozesslandschaft $PL = (A_{PL}, S_{PL}, Z_{PL}, AB_{PL})$ heißt ***Erweiterung von*** $\mathbf{PL}_{top-com}$ ***zu einer kohärenten Prozesslandschaft*** *genau dann, wenn*

1. $A_{PL} = A$
2. *Sei $card(s_i)$ eine Bezeichnung, die angibt, durch wieviele Schnittstellen $\tilde{s} \in S_{PL}$ eine Schnittstelle $s_i \in S$ ersetzt wird.*
   $\forall\, s_i \in S : card(s_i) =$

$$\begin{cases} |\{a \in leaves \mid (a, s_i) \in explicit\}| \text{ falls} \\ \qquad\qquad \{a \in leaves \mid (s_i, a) \in explicit\} = \emptyset \\ |\{a \in leaves \mid (s_i, a) \in explicit\}| \text{ falls} \\ \qquad\qquad \{a \in leaves \mid (a, s_i) \in explicit\} = \emptyset \\ |\{a \in leaves \mid (a, s_i) \in explicit\}| \times \\ |\{a \in leaves \mid (s_i, a) \in explicit\}| \qquad \text{sonst} \end{cases}$$

   Für jede Schnittstelle $s_i \in S$ existiert eine Menge $G(s_i)$ mit

$$G(s_i) = \{s_{i,1}, ..., s_{i,card(s_i)}\},$$
$$G(s_i) \cap S = \emptyset \text{ und}$$
$$\forall\, 1 \leq i, j \leq n, i \neq j : G(s_i) \cap G(s_j) = \emptyset$$

   Damit gilt: $S_{PL} = \dot{\bigcup}_{s_i \in S} G(s_i)$

3. $Z_{PL} = \{(a, \tilde{s}) \mid a \in A, \exists\, s_i : (a, s_i) \in Z, \tilde{s} \in G(s_i)\} \cup$
   $\{(\tilde{s}, a) \mid a \in A, \exists\, s_i : (s_i, a) \in Z, \tilde{s} \in G(s_i)\}$
4. $AB_{PL} = AB$

Mit der ersten Bedingung wird sichergestellt, dass die Menge der Aktivitäten einer Prozesslandschaft im Rahmen der Erweiterung nicht verändert wird. Mit der zweiten Bedingung wird für jede Schnittstelle $s_i \in S$ aus $PL_{top-com}$ die Anzahl der $\tilde{s} \in S_{PL}$ festgelegt, durch die jedes $s_i \in S$ im Rahmen der Erweiterung ersetzt wird. Dazu wird jedem $s_i \in S$ eine Menge $G(s_i) \subset S_{PL}$ zugeordnet, deren Mächtigkeit mit Hilfe der Bezeichnung $card(s_i)$ bestimmt wird:

- Existieren keine lesenden (schreibenden) Zugriffe von Aktivitäten auf eine Schnittstelle $s_i \in S$, entspricht die Mächtigkeit der Menge $G(s_i)$ der Anzahl der schreibenden (lesenden) Zugriffe auf diese Schnittstelle $s_i \in S$.
- Existieren sowohl lesende als auch schreibende Zugriffe auf eine Schnittstelle $s_i \in S$, entspricht die Mächtigkeit von $G(s_i)$ dem Produkt aus der Anzahl der lesenden Zugriffe auf diese Schnittstelle $s_i \in S$ und der Anzahl der schreibenden Zugriffe.

Des Weiteren wird mit $G(s_i) \cap S = \emptyset$ sichergestellt, dass die Menge der Schnittstellen $\tilde{s} \in S_{PL}$ nicht aus der Menge der Schnittstellen $s_i \in S$ konstruiert, sondern neu definiert wird. Die Menge der Schnittstellen $\tilde{s} \in S_{PL}$ lässt sich dann als disjunkte Vereinigung aller Mengen $G(s_i)$ formulieren.

Die dritte Bedingung fordert, dass jeder Zugriff (a, s_i) aus $PL_{top-com}$ so oft als Zugriff $(a, \tilde{s})$ in PL vorkommt, wie Schnittstellen $\tilde{s}$ aus der korrespondierenden Schnittstelle s_i entstanden sind. Dies gilt analog für jeden Zugriff $(s_i, a) \in Z$. Die vierte Bedingung sichert die Beibehaltung des Aktivitätenbaumes AB in unveränderter Form.

Definition 3.2.9. *Sei $PL = (A_{PL}, S_{PL}, Z_{PL}, AB_{PL})$ eine kohärente Prozesslandschaft, die durch Erweiterung der zweiten Ausbaustufe einer Prozesslandschaft $PL_{top-com} = (A, S, Z, AB)$ entstanden ist. Jedes $\tilde{s} \in S_{PL}$ ist durch **Duplizieren** von $s_i \in S = \{s_1, ..., s_n\}$ entstanden:$\Leftrightarrow \tilde{s} \in G(s_i)$.*

Notation 3.2.10. *Wenn keine Missverständnisse auftreten können, wird im Folgenden eine Prozesslandschaft $PL = (A_{PL}, S_{PL}, Z_{PL}, AB_{PL})$ verkürzt als Prozesslandschaft $PL = (A, S, Z, AB)$ bezeichnet.*

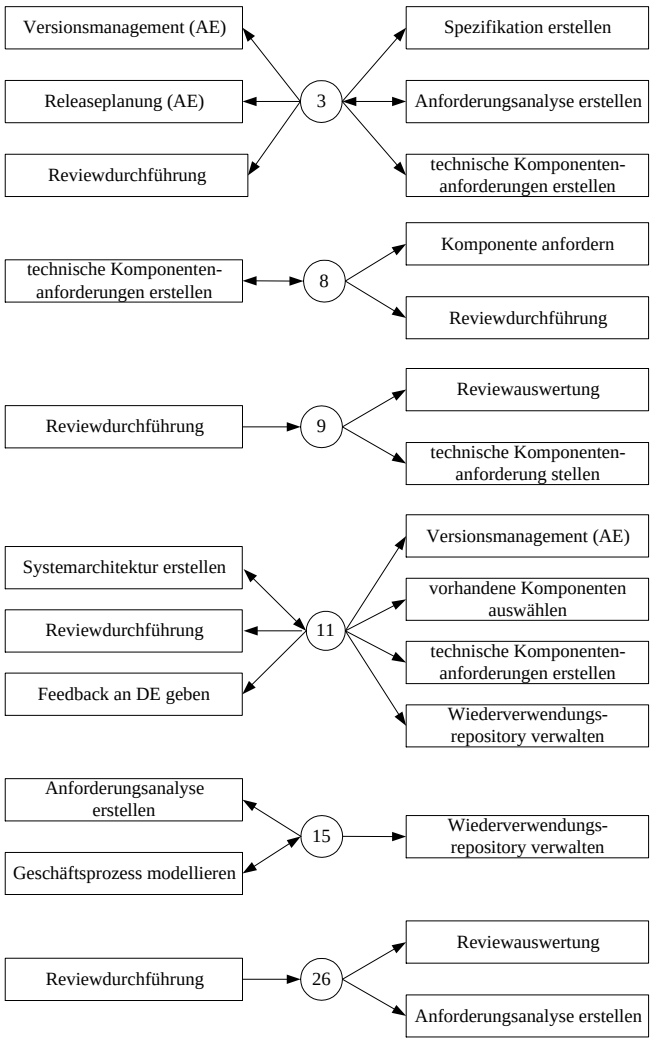
Bemerkung 3.2.11. *Für die Schnittstellen einer kohärenten Prozesslandschaft $PL = (A, S, Z, AB)$ mit $S = S_{intern} \cup S_{extern}$ können weitere Aussagen getroffen werden:*

1. $S_{intern} = \{s \in S \mid \bullet s \cap s\bullet \neq \emptyset\}$
2. $S_{extern} = S \backslash S_{intern}$
3. $|S_{intern}| = |\{a \in A \mid a \in (\bullet s \cap s\bullet)\}|$

Die gemäß Definition 3.2.8 entstandenen Schnittstellen sind immer dann externe Schnittstellen, wenn alle für sie geforderten Einschränkungen gültig sind (vgl. Definition 3.2.4 einer externen Schnittstelle, Seite 96). Es kann jedoch vorkommen, dass auf sie von der gleichen Aktivität sowohl lesend als auch schreibend zugegriffen wird, ihr Vor- und Nachbereich also gemeinsame Elemente haben. Ist $\bullet s \cap s\bullet \neq \emptyset$, so handelt es sich um eine interne Schnittstelle. Ihre Anzahl entspricht der Anzahl an Aktivitäten, die sowohl im Vor- als auch im Nachbereich einer Schnittstelle s enthalten ist.

Abbildung 3.12 zeigt die Dokumentensicht der im Beispiel beteiligten Schnittstellen. Alle Aktivitäten, die zusätzlich zu den ursprünglich betrachteten in Relation zu diesen Schnittstellen stehen (vgl. Abbildung 3.11), werden im Rahmen des ersten Erweiterungsschrittes ebenfalls berücksichtigt. Konkret sind dies die Aktivitäten *Versionsmanagement (AE)*, *Releaseplanung (AE)*, *Reviewdurchführung (AE)*, *Spezifikation erstellen*, *Komponente anfordern*, *Wiederverwendungsrepository verwalten*, *Projekt einschätzen* und *Reviewauswertung*. Die Abkürzung (AE) zeigt an, dass die damit versehenen Aktivitäten verfeinernde Aktivitäten des Application Engineering sind.

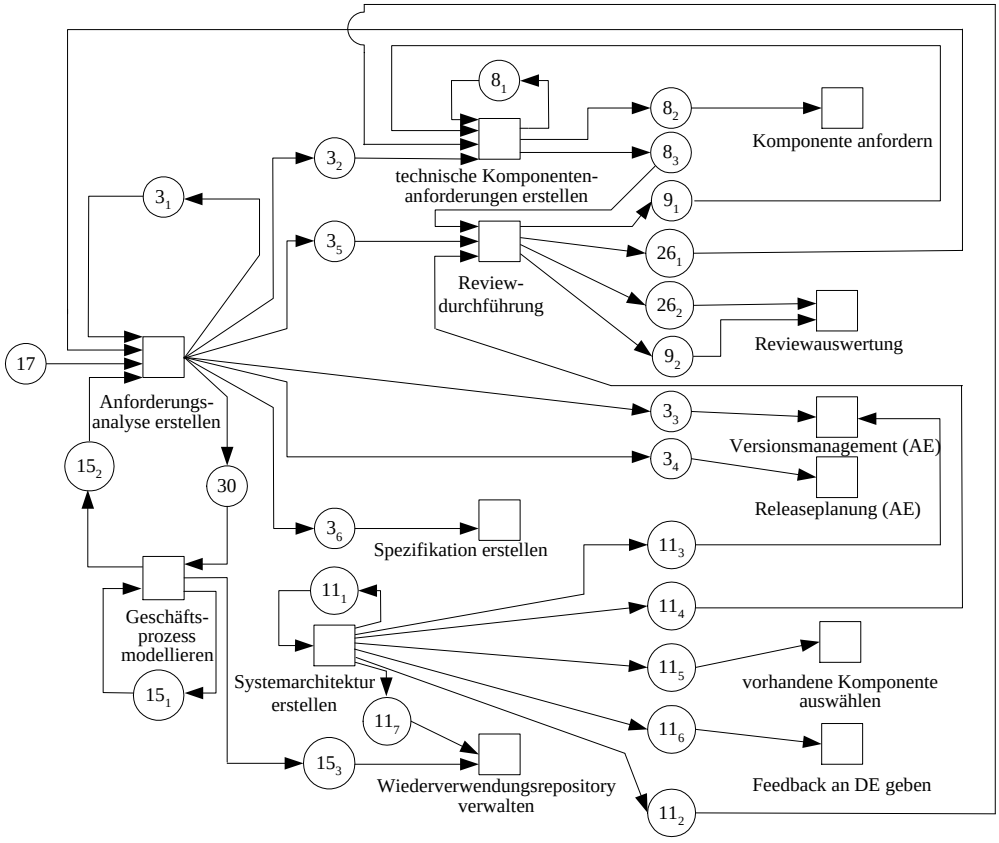
Für den in Abbildung 3.11 vorgestellten Ausschnitt der Beispielprozesslandschaft liefert der erste Erweiterungsschritt das in Abbildung 3.13 dargestellte Ergebnis. Um die Verständlichkeit trotz steigender Komplexität der dargestellten Prozesslandschaft weiter zu gewährleisten, werden die Namen der Aktivitäten nachfolgend unterhalb der sie



3 Anforderungsdokument
8 technische Komponentenanforderung
9 Review technische Komponentenanforderung
11 Systemarchitektur
15 Geschäftsprozessmodell
26 Review Anforderungsdokument

Abbildung 3.12: Dokumentensicht der Schnittstellen *Anforderungsdokument*, *technische Komponentenanforderung*, *Review technische Komponentenanforderung*, *Systemarchitektur*, *Geschäftsprozessmodell* und *Review Anforderungsdokument*

repräsentierenden Rechtecke aufgeführt, sofern sie nicht nur aus einem Buchstaben bestehen. Die Schnittstellen behalten ihre Nummern bei, werden jedoch indiziert. So bleibt transparent, welche Schnittstelle in PL durch die Duplizierung welcher Schnittstelle aus $PL_{top-com}$ entstanden ist.



3_1,..., 3_6	Anforderungsdokument
8_1, 8_2 , 8_3	technische Komponentenanforderung
9_1, 9_2	Review technische Komponentenanforderung
11_1,..., 11_7	Systemarchitektur
15_1,15_2, 15_3	Geschäftsprozessmodell
17	erste Anforderungen
26_1, 26_2	Review Anforderungsdokument
30	Anfrage nach Geschäftsprozessmodell

Abbildung 3.13: Kohärente Prozesslandschaft als Ergebnis des ersten Erweiterungsschrittes

Im vorliegenden Beispiel liegt die Schnittstelle *Anforderungsdokument* jetzt sechsmal vor, die Schnittstelle *technische Komponentenanforderung* dreimal, *Systemarchitektur* siebenmal, *Geschäftsprozessmodell* dreimal und *Review Anforderungsdokument* sowie *Review technische Komponentenanforderung* jeweils zweimal.

Durch die Duplizierung der Schnittstellen und einer geeigneten Verknüpfung von Ak-

tivitäten und Schnittstellen über Zugriffe (gemäß Definition 3.2.8) entsteht eine kohärente Prozesslandschaft PL, in der alle Informationen über mögliche Datenflüsse der Prozesslandschaft $PL_{top-com}$ enthalten sind. Sie liefert die Basis für den zweiten Erweiterungsschritt, der der Landschaft Informationen über das Ablaufverhalten in Form von Verhaltensvorschriften hinzufügt.

3.2.1.2 Festlegung der Schaltverhalten

Um Informationen über das dynamische Verhalten einer Prozesslandschaft analysieren zu können, muss das Schaltverhalten (Schaltfähigkeit und Schaltregeln) ihrer Aktivitäten betrachtet werden. Dies geschieht durch das Festlegen von Bedingungen für die Belegungen des Vor- und Nachbereichs der zu betrachtenden Aktivitäten mit Marken vor und nach dem Schalten. Marken repräsentieren Informationsobjekte, die innerhalb einer Prozesslandschaft von schaltfähigen Aktivitäten verarbeitet bzw. erzeugt werden. Mit Hilfe von Marken kann die Ausführung eines Petrinetzes definiert werden.

Bemerkung 3.2.12. *Über das konkrete Schaltverhalten in einer kohärenten Prozesslandschaft kann zunächst keine Aussage getroffen werden. In einfachen Petrinetzen wie Bedingungs-/Ereignisnetzen und Stellen-/Transitionsnetzen wird ein sogenanntes einfaches Schaltverhalten verwendet, d.h. eine Aktivität ist schaltfähig, wenn alle Schnittstellen in ihrem Vorbereich jeweils mit mindestens einer Marke belegt sind und die Kapazität aller Ausgabeschnittstellen ausreichend ist. Für Kanten ohne Gewichtsangabe wird dabei üblicherweise ein Kantengewicht von 1 angenommen; Stellen ohne Kapazitätsangabe wird eine Kapazität von ∞ zugeordnet [Bau96]. Die Schaltung einer Aktivität bewirkt die Belegung aller Schnittstellen in ihrem Nachbereich mit einer (weiteren) Marke.*

In einer kohärenten Prozesslandschaft (vgl. Bemerkung 3.1.29 und Definition 3.2.3) entstehen viele Ablaufsituationen, in denen nur eine Teilmenge aller Schnittstellen im Vorbereich einer Aktivität mit Marken belegt sein muss, um eine Schaltung zu erlauben. Des Weiteren können Ablaufsituationen auftreten, bei denen die Schaltung einer Aktivität lediglich die Belegung einer Teilmenge aller Schnittstellen in ihrem Nachbereich bewirken soll. Für die verschiedenen Ablaufsituationen werden den Aktivitäten einer Prozesslandschaft verschiedene Aktivierungsbedingungen und Regeln zur Folgemarkierung von Schnittstellen im Nachbereich zugeordnet. Für diesen Zweck werden u.a. die für Petrinetze bereits bekannten Arten des deterministischen und komplexen Schaltverhaltens [Gru91] verwendet. Während diese bei Gruhn jedoch als Bestandteil von Jobs definiert sind (die wiederum Aktivitäten eines Softwareprozesses ausführen), werden sie im Rahmen des Process Landscaping über ausgezeichnete Teilmengen der Vor- und Nachbereiche einer Aktivität festgelegt, die je nach Ablaufsituation für ihre Schaltfähigkeit und Schaltung genutzt werden. Es gibt mehrere Gründe für diese veränderte Definition. Zum einen erleichtert sie die Einführung einer weiteren Art von Schaltverhalten, die gleichzeitig auch mögliche Abhängigkeiten zwischen Teilmengen der Vor- und Nachbereiche einer Aktivität berücksichtigt (vgl. Definition 3.2.26). Derartige Ablaufsituationen weisen einen Modellierer auf sinnvolle Stellen zur Verfeine-

rung der Prozesslandschaft hin (vgl. Abschnitt 3.2.1.4). Zum anderen müssen Ablaufsituationen, an denen externe Schnittstellen beteiligt sind, gesondert betrachtet werden, um Konfliktsituationen zu vermeiden, die Synchronieaussagen erschweren oder sogar verhindern. Auch dies fällt leichter, wenn bei der Definition eines Schaltverhaltens die beteiligten Schnittstellen aus dem Vor- und Nachbereich einer betrachteten Aktivität im Vordergrund stehen (vgl. Abbildungen 3.15 und 3.17 auf den Seiten 106 und 107).

Notation 3.2.13. *Im Folgenden sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz und $M : S \to \mathbb{N}$ eine Markierungsfunktion (kurz: Markierung), die jeder Schnittstelle $s \in S$ eine Anzahl von Marken zuordnet.*

Bemerkung 3.2.14. *Üblicherweise wird bei der Definition von markierten Netzen diesem immer eine feste Startmarkierung M_0 zugeordnet. Da diese im Rahmen des Process Landscaping jedoch nicht benötigt wird, wird in den nachfolgenden Definitionen darauf verzichtet.*

Definition 3.2.15. *Sei $S_{source,a} \subseteq {\bullet}a$ eine Teilmenge des Vorbereichs einer Aktivität $a \in A_{PN}$. Die Aktivität $a \in A_{PN}$ heißt **schaltfähig** :⇔*

$$\forall\, s \in S_{source,a} : M(s) \geq 1 \text{ und}$$
$$S_{source,a} \neq \emptyset$$

Definition 3.2.16. *Eine Teilmenge $S_{source,a} \subseteq {\bullet}a$ für ein $a \in A_{PN}$ heißt*

Schaltmenge von a *:⇔ a ist schaltfähig.*

*Die Menge $(H_{source,a})$ legt – mit der Menge aller möglichen Teilmengen des Vorbereichs einer Aktivität als Elemente – die **Menge aller Schaltmengen** einer Aktivität $a \in A_{PN}$ fest:*

$$(H_{source,a}) := \left\{ S_{source,a} \mid S_{source,a} \text{ ist Schaltmenge von } a \right\}$$

Definition 3.2.17. *Sei $\mathcal{M}(\mathbb{N})$ die Menge aller Markierungen eines kohärenten Petrinetzes und M eine feste Markierung. **Eine Aktivität** $a \in A_{PN}$ **schaltet von** $\mathbf{M}$ **nach** $\mathbf{M'}$, wenn a unter M schaltfähig ist und M' aus M durch Entnahme von Marken aus den Schnittstellen des Vorbereichs von a und Ablage von Marken auf Schnittstellen des Nachbereichs von a entsteht.* $\mathbf{M'}$ ***heißt Folgemarkierung von*** $\mathbf{M}$.

Definition 3.2.18. *Eine Teilmenge $S_{dest,a} \subseteq a{\bullet}$ für ein $a \in A_{PN}$ heißt **Ergebnismenge von a** :⇔*

$$\forall\, s' \in a{\bullet} : M'(s') = \begin{cases} M(s') + 1 & \text{falls } s' \in S_{dest,a} \\ M(s') & \text{sonst} \end{cases} \quad \text{und}$$
$$S_{dest,a} \neq \emptyset$$

*Die Menge $(H_{dest,a})$ legt – mit der Menge aller möglichen Teilmengen des Nachbereichs einer Aktivität als Elemente – die **Menge aller Ergebnismengen** einer Aktivität $a \in A_{PN}$ fest:*

$$(H_{dest,a}) := \{S_{dest,a} \mid S_{dest,a} \text{ ist Ergebnismenge von } a\}$$

Definition 3.2.19. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **einfaches Eingangsschaltverhalten** :⇔*

1. $|\bullet a| \geq 1$
2. $\forall\, S_{source,a} \in (H_{source,a}) : |S_{source,a}| = |\bullet a\,|$
3. $\forall\, s \in \bullet a : M'(s) = M(s) - 1$

Eine Aktivität mit einem einfachen Eingangsschaltverhalten hat somit genau eine Schaltmenge. Diese Schaltmenge ist mit dem Vorbereich der betrachteten Aktivität identisch.

Definition 3.2.20. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **einfaches Ausgangsschaltverhalten** :⇔*

1. $|a\bullet| \geq 1$
2. $\forall\, S_{dest,a} \in (H_{dest,a}) : |S_{dest,a}| = |a\bullet|$
3. $\forall\, s \in a\bullet : M'(s) = M(s) + 1$

Hat eine Aktivität ein einfaches Ausgangsschaltverhalten, hat sie genau eine Ergebnismenge. Diese Ergebnismenge ist mit dem Nachbereich der betrachteten Aktivität identisch.

Definition 3.2.21. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **deterministisches Eingangsschaltverhalten** :⇔*

1. $|\bullet a| \geq 2$
2. $\forall\, S_{source,a} \in (H_{source,a}) : |S_{source,a}| = 1$
3. $\forall\, s \in \bullet a\ \exists!\, S_{source,a}$ *mit* $s \in S_{source,a}$*:* $M'(s) = \begin{cases} M(s) - 1 & \textit{falls } s \in S_{source,a} \\ M(s) & \textit{sonst} \end{cases}$

Die Zuordnung des deterministischen Eingangsschaltverhaltens ist eingeschränkt auf diejenigen Aktivitäten, deren Vorbereich mehr als eine Schnittstelle umfasst. Zusätzlich wird gefordert, dass jede Schaltmenge genau eine Schnittstelle beinhaltet und somit die Anzahl der verschiedenen Schaltmengen mit der Anzahl der Schnittstellen im Vorbereich der korrespondierenden Aktivität übereinstimmt.

Ein deterministisches Eingangsschaltverhalten einer Aktivität lässt sich innerhalb eines Petrinetzes auch wie in Abbildung 3.14 darstellen [Gru91]. Die Bezeichnung XOR

kennzeichnet dort das deterministische Eingangsschaltverhalten der Aktivität a_3, und die mit s_i bezeichneten Schnittstellen repräsentieren jeweils interne Schnittstellen. In der äquivalenten Darstellung (vgl. rechte Seite der Abbildung 3.14) haben die Aktivitäten a_1 und a_2 eine gemeinsame Ergebnismenge.

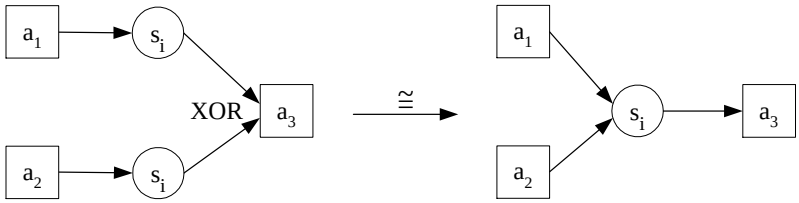


Abbildung 3.14: Deterministisches Eingangsschaltverhalten bei internen Schnittstellen s_i im Vorbereich einer Aktivität a_3

Diese Form einer äquivalenten Darstellung ist nur für interne Schnittstellen gültig. Sie ist für externe Schnittstellen weiter anzupassen, da ansonsten die Restriktionen für den Vorbereich von externen Schnittstellen verletzt werden, für den gilt, dass er nur genau eine Aktivität beinhalten darf (vgl. Definition 3.2.4, Seite 96). Abbildung 3.15 zeigt diese Anpassung, bei der zwei Empfängeraktivitäten $a_{recipient-1}$, $a_{recipient-2}$ und eine interne Schnittstelle s_i derart zwischengefügt sind, dass $a_{recipient-1}$ den Nachbereich der oberen externen Schnittstelle s_e und $a_{recipient-2}$ den Nachbereich der unteren externen Schnittstelle s_e bildet. Die beiden Empfängeraktivitäten haben das gleiche Eingangsschaltverhalten wie die Aktivitäten a_1 und a_2 in Abbildung 3.14.

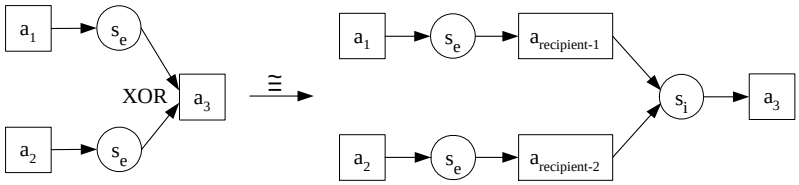


Abbildung 3.15: Deterministisches Eingangsschaltverhalten bei externen Schnittstellen s_e im Vorbereich einer Aktivität a_3

Definition 3.2.22. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **deterministisches Ausgangsschaltverhalten** :⇔*

1. $|a\bullet| \geq 2$
2. $\forall\, S_{dest,a} \in (H_{dest,a}) : |S_{dest,a}| = 1$
3. $\forall\, s \in a\bullet\ \exists!\, S_{dest,a}$ *mit* $s \in S_{dest,a}$: $M'(s) = \begin{cases} M(s)+1 & \textit{falls } s \in S_{dest,a} \\ M(s) & \textit{sonst} \end{cases}$

Die Restriktionen für die Ergebnismengen bei einem deterministischen Ausgangsschaltverhalten sind analog zu denen für die Schaltmengen bei einem deterministischen Eingangsschaltverhalten.

Durch ein deterministisches Eingangsschaltverhalten lässt sich eine Ablaufsituation modellieren, bei der aus dem Vorbereich einer schaltfähigen Aktivität genau aus einer von mehreren Schnittstellen eine Marke entnommen wird. Analog wird durch ein deterministisches Ausgangsschaltverhalten nur genau eine von mehreren vorhandenen Schnittstellen des Nachbereichs mit einer (weiteren) Marke belegt. Welcher der Schnittstellen im Nachbereich von a durch die Folgemarkierung M' eine Marke zugeordnet wird, kann zufällig oder durch weitere Bedingungen festgelegt werden.

Ein deterministisches Ausgangsschaltverhalten einer Aktivität a_1 lässt sich äquivalent auch als Konflikt darstellen [Gru91]. Abbildung 3.16 verdeutlicht diese Situation für interne Schnittstellen innerhalb einer Prozesslandschaft.

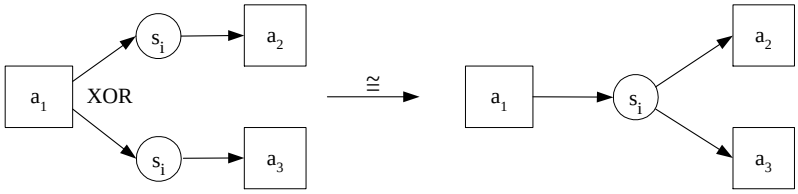


Abbildung 3.16: Deterministisches Ausgangsschaltverhalten bzgl. interner Schnittstellen im Nachbereich einer Aktivität a_1

Diese Form einer äquivalenten Darstellung ist nur für interne Schnittstellen einer Prozesslandschaft gültig. Sie ist für externe Schnittstellen weiter anzupassen, da sonst analog zum Fall des deterministischen Eingangsschaltverhaltens die Restriktionen für den Nachbereich von externen Schnittstellen verletzt werden. Die rechte Seite der Abbildung 3.17 zeigt diese Anpassung, bei der, ähnlich dem Fall des deterministischen Eingangsschaltverhaltens, eine interne Schnittstelle s_i und zwei Senderaktivitäten $a_{sender-1}, a_{sender-2}$ als Nachbereich von s_i derart zwischengefügt sind, dass die obere externe Schnittstelle s_e die Ergebnismenge von $a_{sender-1}$ bildet und die untere externe Schnittstelle s_e die Ergebnismenge von $a_{sender-1}$. Beide Senderaktivitäten haben das gleiche Eingangsschaltverhalten wie die Aktivitäten a_2 und a_3 in Abbildung 3.16.

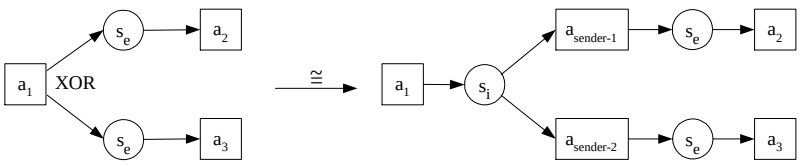


Abbildung 3.17: Deterministisches Ausgangsschaltverhalten bzgl. externer Schnittstellen im Nachbereich einer Aktivität a_1

Definition 3.2.23. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **komplexes Eingangsschaltverhalten** :⇔*

1. $|\bullet a| \geq 2$
2. $\forall\, S_{source,a} \in (H_{source,a}) : |S_{source,a}| \geq 1 \wedge |(H_{source,a})| > 1$
3. $\forall\, s \in \bullet a\ \exists!\, S_{source,a}$ *mit* $s \in S_{source,a}$: $M'(s) = \begin{cases} M(s) - 1 & \textit{falls } s \in S_{source,a} \\ M(s) & \textit{sonst} \end{cases}$

Analog zum deterministischen wird auch für das komplexe Eingangsschaltverhalten gefordert, dass der Vorbereich der korrespondierenden Aktivität mehr als eine Schnittstelle umfasst. Zusätzlich dürfen bei einem komplexen Eingangsschaltverhalten die Schaltmengen mehr als eine Schnittstelle enthalten, und es müssen mindestens zwei verschiedene, aber nicht notwendigerweise disjunkte Schaltmengen existieren.

Definition 3.2.24. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein **komplexes Ausgangsschaltverhalten** :⇔*

1. $|\, a\bullet\,| \geq 2$
2. $\forall\, S_{dest,a} \in (H_{dest,a}) : |S_{dest,a}| \geq 1 \wedge |(H_{dest,a})| > 1$
3. $\forall\, s \in a\bullet\ \exists!\, S_{dest,a}$ *mit* $s \in S_{dest,a} : M'(s) = \begin{cases} M(s) + 1 & \textit{falls } s \in S_{dest,a} \\ M(s) & \textit{sonst} \end{cases}$

Bei einem komplexen Schaltverhalten einer Aktivität muss immer eine von mehreren vorhandenen Schaltmengen im Vorbereich markiert sein bzw. eine von mehreren vorhandenen Ergebnismengen im Nachbereich durch eine Folgemarkierung belegt werden. Ein deterministisches Schaltverhalten stellt somit einen Spezialfall des komplexen Schaltverhaltens dar. Jedes komplexe Eingangsschaltverhalten lässt sich jedoch auch durch eine Folge von einfachen Eingangsschaltverhalten ausdrücken. Gleiches gilt für das komplexe Ausgangsschaltverhalten. Diese Rückführung vom komplexen auf das einfache Schaltverhalten wird im Folgenden näher erläutert.

Abbildung 3.18 zeigt links eine Aktivität A mit komplexem Eingangsschaltverhalten. Die verschiedenen vorhandenen Schaltmengen sind durch die gestrichelten Linien erkennbar. Insgesamt hat Aktivität A vier Schnittstellen in ihrem Vorbereich, die sich in den vier verschiedenen Schaltmengen $\{a\}$, $\{b, c\}$, $\{c, d\}$ und $\{d\}$ wiederfinden. Letztere sind so konstruiert, dass alle möglichen Kombinationen von Schaltmengen vorkommen, nämlich

- Schaltmengen, die nur eine einzelne Schnittstelle enthalten,
- Schaltmengen, die Teilmenge einer anderen Schaltmenge sind und
- Schaltmengen, die gemeinsame Schnittstellen haben.

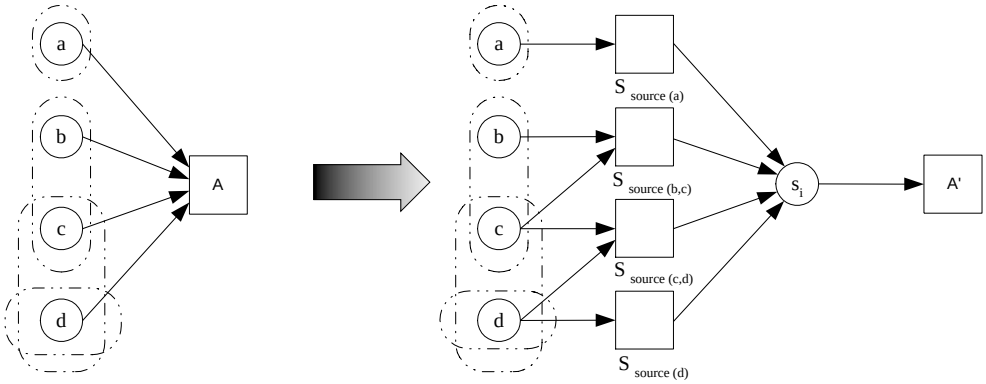


Abbildung 3.18: Rückführung eines komplexen auf ein einfaches Eingangsschaltverhalten

Die rechte Seite der Abbildung 3.18 zeigt, wie das komplexe Eingangsschaltverhalten der Aktivität A durch eine Menge von Aktivitäten S_i mit jeweils einfachem Ein- und Ausgangsschaltverhalten dargestellt ist. Pro Schaltmenge der Aktivität A ist dort eine Aktivität $S_{source(a)}$, $S_{source(b,c)}$, $S_{source(c,d)}$ und $S_{source(d)}$ mit einfachem Ein- und Ausgangsschaltverhalten modelliert, die genau eine der Schaltmengen als Vorbereich und eine gemeinsame (neue) Schnittstelle $s_i \in S_{intern}$ als Nachbereich hat. Diese Schnittstelle s_i bildet den Vorbereich der Aktivität A', die damit ebenfalls ein einfaches Eingangsschaltverhalten aufweist. Die Aktivitäten A und A' unterscheiden sich damit lediglich durch ihre unterschiedlichen Vorbereiche; semantisch sind sie als identisch anzusehen.

Durch die Veränderung des Schaltverhaltens der Aktivität A mit Hilfe der Aktivitäten $S_{source(a)}$, $S_{source(b,c)}$, $S_{source(c,d)}$ und $S_{source(d)}$ werden die Schaltmengen von A teilweise den Vorbereichen mehrerer Aktivitäten zugeordnet. Diese Situation tritt immer dann auf, wenn eine der beteiligten Schnittstellen in mehr als einer Schaltmenge enthalten ist. Sind a, b, c und d externe Schnittstellen, so ist durch die Veränderung die Restriktion gemäß Definition 3.2.4 auf Seite 96 verletzt worden, nach der externe Schnittstellen nur genau eine Aktivität in ihrem Nachbereich haben dürfen. In diesem Fall muss die in Abbildung 3.18 dargestellte Situation analog der in Abbildung 3.15 auf Seite 106 dargestellten Erweiterung mittels der Empfängeraktivitäten $a_{recipient-1}$, $a_{recipient-2}$ und einer zusätzlichen internen Schnittstelle im Vorbereich der betreffenden Schnittstelle angepasst werden.

Wie sich das komplexe Ausgangsschaltverhalten einer Aktivität A durch eine Folge von einfachen Ausgangsschaltverhalten ausdrücken lässt, ist in Abbildung 3.19 dargestellt. Diese hat die Ergebnismengen $\{a, b\}$, $\{b, c\}$ und $\{c, d\}$ in ihrem Nachbereich. Analog zur Verfeinerung einer Aktivität mit komplexem Eingangsschaltverhalten werden hier pro Ergebnismenge eine verfeinernde Aktivität $S_{dest(a,b)}$, $S_{dest(b,c)}$ und $S_{dest(c,d)}$ mit einfachem Ein- und Ausgangsschaltverhalten modelliert, wobei jede Aktivität genau eine der Ergebnismengen als Nachbereich und eine gemeinsame

(neue) Schnittstelle $s_i \in S_{intern}$ als Vorbereich hat. Die Aktivitäten A und A' sind wiederum als semantisch identisch anzusehen.

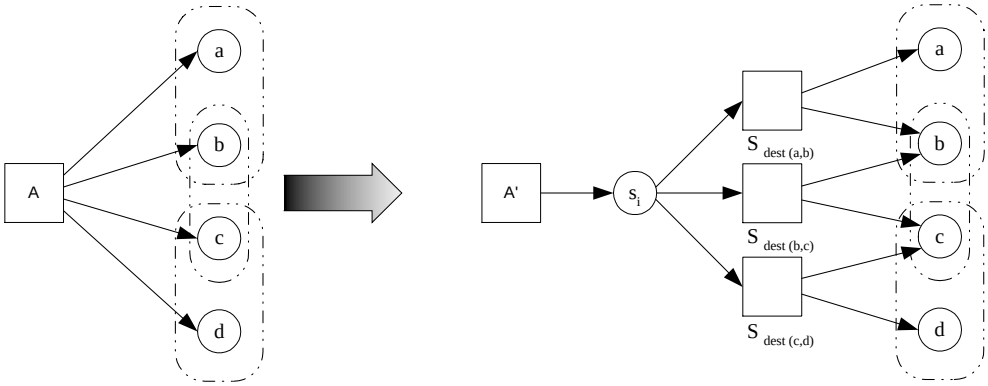


Abbildung 3.19: Rückführung eines komplexen auf ein einfaches Ausgangsschaltverhalten

Falls $\{a\}$, $\{b\}$, $\{c\}$ und/oder $\{d\}$ externe Schnittstellen sind, können durch die Rückführung auch hier Situationen entstehen, die die Restriktionen aus Definition 3.2.4 verletzen. In diesen Fällen ist das Petrinetz entsprechend der in Abbildung 3.17 dargestellten Erweiterung mittels der Senderaktivitäten $s_{sender-1}$, $s_{sender-2}$ und einer zusätzlichen internen Schnittstelle im Vorbereich der betreffenden Schnittstelle anzupassen.

Das Wissen um die Rückführung eines komplexen auf ein einfaches Schaltverhalten unterstützt einen Modellierer in zweierlei Hinsicht. Zum einen ist es bei der Verfeinerung einer Aktivität mit komplexem Schaltverhalten hilfreich, zum anderen soll ein Modellierer bei der Erstellung der unteren Ebenen einer Prozesslandschaft nicht auf diejenigen Petrinetz-Varianten beschränkt werden, die die Modellierung von komplexen Schaltverhalten unterstützen. Für letzteren Fall ist über die diskutierte Rückführung eine Möglichkeit gegeben, nach dem Hinzufügen von ersten Ablaufinformationen auch ohne die verschiedenen Schaltverhalten eine Prozesslandschaft detaillierter modellieren zu können.

Definition 3.2.25. *Sei $a \in A_{PN}$ eine Aktivität, für die gilt, dass $|(H_{source,a})| > 1$ und $|(H_{dest,a})| > 1$. Sei weiterhin $DEP_a \subseteq (H_{source,a}) \times (H_{dest,a})$ Teilmenge des Kreuzproduktes der Schalt- und Ergebnismengen der Aktivität a. Die Relation $\mathbf{DEP_a}$ definiert Abhängigkeiten zwischen der Belegung von Schalt- und Ergebnismengen einer Aktivität a :⇔*

$$\forall\, S_{source,a} \in (H_{source,a}) \,\exists\, S_{dest,a} \in (H_{dest,a}) : (S_{source,a}, S_{dest,a}) \in DEP_a$$

Über DEP_a wird für eine Aktivität $a \in A_{PN}$ festgelegt, welche Ergebnismenge $S_{dest,a} \in (H_{dest,a})$ bei der Belegung welcher Schaltmenge $S_{source,a} \in (H_{source,a})$ nach dem Schalten mit Marken belegt wird.

Definition 3.2.26. *Eine schaltfähige Aktivität $a \in A_{PN}$ hat ein von der Belegung ihres Vorbereiches* ***abhängiges Ausgangsschaltverhalten*** $:\Leftrightarrow$

1. $|a\bullet| > 1 \wedge |\bullet a| > 1$
2. $\forall\, S_{dest,a} \in (H_{dest,a}) : |S_{dest,a}| \geq 1 \wedge |(H_{dest,a})| > 1$
3. $\forall\, S_{source,a} \in (H_{source,a}) : |S_{source,a}| \geq 1 \wedge |(H_{source,a})| > 1$
4. $\forall\, s \in a\bullet\ \exists!\, S_{dest,a}$ *mit* $s \in S_{dest,a} : M'(s) = \begin{cases} M(s)+1 & \textit{falls } s \in S_{dest,a} \\ M(s) & \textit{sonst} \end{cases}$
5. $\forall\, S_{source,a} \in (H_{source,a})\, \exists\, S_{dest,a} \in (H_{dest,a}) : (S_{source,a}, S_{dest,a}) \in DEP_a$
6. $|DEP_a| > 1$

Die Bedingungen 1. bis 4. fordern ein komplexes Ein- und Ausgangsschaltverhalten der Aktivität a. Des Weiteren muss jeder Schaltmenge genau eine Ergebnismenge zugeordnet werden, die nach dem Schalten der Aktivität a mit einer (weiteren) Marke belegt wird (Bedingung 5.). Außerdem wird mit Bedingung 6. gefordert, dass DEP_a mindestens zwei Elemente enthält.

Für die Betrachtung des dynamischen Verhaltens einer Prozesslandschaft mit Hilfe der Ein- und Ausgangsschaltverhalten aller Aktivitäten ist gemäß der Definitionen 3.2.19 bis 3.2.26 grundsätzlich immer festzulegen, welche Teilmengen aus dem Vor- bzw. Nachbereich mögliche Schalt- bzw. Ergebnismengen sind. Da hierfür keine allgemeingültigen und kontextunabhängigen Aussagen gemacht werden können, muss ein Modellierer in Abhängigkeit der modellierten Situationen das jeweilige Schaltverhalten (neu) definieren. Auf der Petrinetzebene ergibt sich somit in Abhängigkeit der Anzahl von Schaltmengen einer Aktivität und ihren Relationen zueinander eine konkrete Form des Eingangsschaltverhaltens. Analoges gilt für das Ausgangsschaltverhalten.

Definition 3.2.27. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz. Die Abbildung $\mathbf{f_{source}}$ ordnet jeder Aktivität $a \in A_{PN}$ ein Eingangsschaltverhalten zu:*

$$f_{source} : A_{PN} \rightarrow \{all, det, complex\}$$

Dabei bezeichnet all das einfache, det das deterministische und $complex$ das komplexe Schaltverhalten.

Definition 3.2.28. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz. Die Abbildung $\mathbf{f_{dest}}$ ordnet jeder Aktivität $a \in A_{PN}$ ein Ausgangsschaltverhalten zu:*

$$f_{dest} : A_{PN} \rightarrow \{all, det, complex, dependent\}$$

Dabei bezeichnet $dependent$ das abhängige Schaltverhalten.

Mit den Markierungsfunktionen M und M' sowie den Abbildungen f_{source} und f_{dest} ist das einer Prozesslandschaft zugrundeliegende Petrinetz um Eigenschaften ergänzt worden, die die Beschreibung dynamischen Verhaltens erlauben. Damit führen diese Ergänzungen zur nächsten Ausbaustufe einer Prozesslandschaft.

Definition 3.2.29. *Sei $PL_{top-com} = (A, S, Z, AB)$ die zweite Ausbaustufe einer Prozesslandschaft und $PL_{top-com}$ kohärent. Seien weiterhin*

- $M : S \to \mathbb{N}$ *und* $M' : S \to \mathbb{N}$ *Markierungsfunktionen, die jeder Schnittstelle eine Anzahl von Marken zuordnen,*
- f_{source} *eine Abbildung zur Festlegung des Eingangsschaltverhaltens gemäß Definition 3.2.27 und*
- f_{dest} *eine Abbildung zur Festlegung des Ausgangsschaltverhaltens gemäß Definition 3.2.28.*

*Die **dritte Ausbaustufe einer Prozesslandschaft** ist definiert als ein Tupel*

$$PL_{bottom} := (PL_{top-com}, M, M', f_{source}, f_{dest})$$

Bemerkung 3.2.30. *Der Index* bottom *der Prozesslandschaft PL_{bottom} deutet an, dass es sich um eine Prozesslandschaft handelt, deren untere Ebenen im Mittelpunkt der Modellierung stehen.*

Notation 3.2.31. *Wenn keine Missverständnisse auftreten können, wird im Folgenden eine Prozesslandschaft $PL_{bottom} = (PL_{top-com}, M, M', K, f_{source}, f_{dest})$ verkürzt als Prozesslandschaft $PL_{bottom} = (A, S, Z, AB)$ bezeichnet.*

Nach der Zuordnung der Schaltverhalten zu allen Aktivitäten einer Prozesslandschaft können aus Gründen der Übersichtlichkeit noch diejenigen Schnittstellen inklusive ihrer zugehörigen Zugriffe entfernt werden, die bei keinem Schaltverhalten Berücksichtigung gefunden haben und somit in keiner Ergebnismenge liegen.

Definition 3.2.32. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz. Sei weiter*

$$(H_{dest,A_{PN}}) := \bigcup_{a \in A_{PN}} (H_{dest,a})$$

*die Menge aller Ergebnismengen eines kohärenten Petrinetzes. Eine Schnittstelle $s \in S$ heißt **relevant**:$\Leftrightarrow$*

$$s\bullet = \emptyset \vee \bullet s = \emptyset \vee \exists\, S_{dest,a} \in (H_{dest,A_{PN}}) : s \in S_{dest,a}$$

Eine Schnittstelle wird also als relevant bezeichnet, wenn sie in mindestens einer Ergebnismenge liegt oder ihr Vor- oder Nachbereich leer ist. Im letzten Fall wird s als Randstelle bezeichnet.

Definition 3.2.33. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz. Die Menge*

$$S_{relevant} := \{s \mid s \text{ ist relevant}\}$$

bezeichnet die Menge aller relevanten Schnittstellen innerhalb eines kohärenten Petrinetzes $PN = (A_{PN}, S, Z_{PN})$.

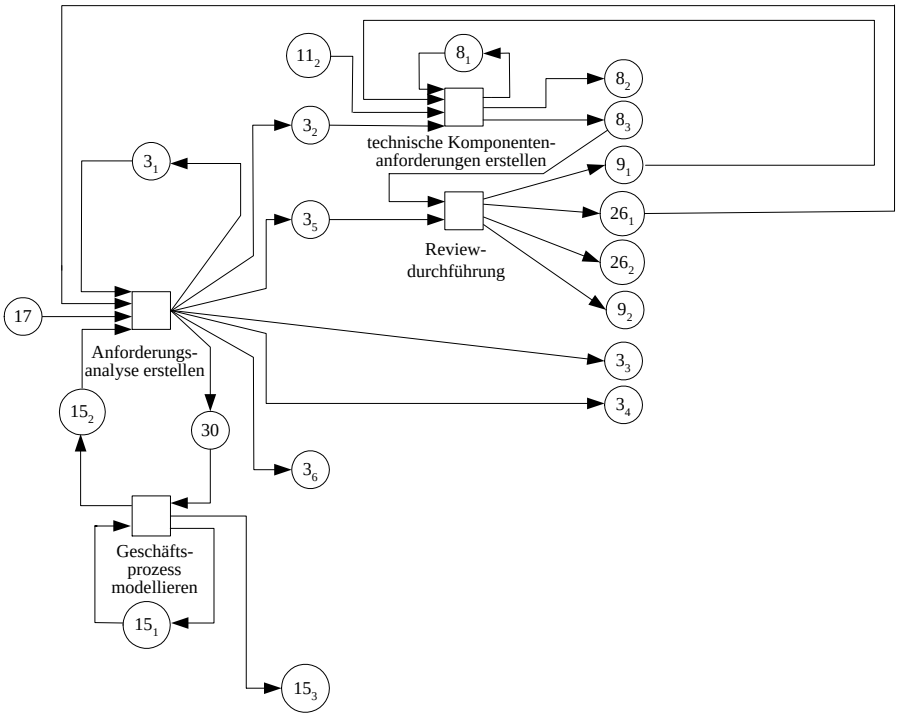
Die Menge der irrelevanten Schnittstellen enthält diejenigen Schnittstellen, die durch die Konstruktion des kohärenten Petrinetzes entstanden sind, um die Darstellung aller möglichen Datenflüsse sicherzustellen, die aber in keiner realen Ablaufsituation benötigt werden, also weder in einer Schalt- noch in einer Ergebnismenge enthalten sind. Werden sie inklusive ihrer zugehörigen Zugriffe wieder aus einer Prozesslandschaft entfernt, können sich für die korrespondierenden Aktivitäten Änderungen der Schaltverhalten ergeben. Aus einem komplexen Schaltverhalten kann so beispielsweise ein einfaches oder ein deterministisches Schaltverhalten werden. Die Eliminierung der irrelevanten Schnittstellen ist nicht zwingend erforderlich, da die korrespondierenden Ablaufsituationen bei einer Analyse nicht berücksichtigt werden. Dieser Schritt erleichtert jedoch die Analyse durch die Verringerung der Komplexität der dargestellten Prozesslandschaft.

In Tabelle 3.4 sind die Ein- und Ausgangsschaltverhalten der Aktivitäten *Anforderungsanalyse erstellen*, *Geschäftsprozess modellieren*, *technische Komponentenanforderungen erstellen* und *Reviewdurchführung* aus der Beispielprozesslandschaft (vgl. Abbildung 3.13 auf Seite 102) aufgeführt. Alle weiteren in Abbildung 3.13 abgebildeten Aktivitäten werden in der nachfolgenden Diskussion nicht weiter berücksichtigt. Den jeweiligen Schaltverhalten sind die konkreten Schalt- und Ergebnismengen zugeordnet. Bei einem abhängigen Ausgangsschaltverhalten sind zusätzlich die Elemente $dep_i \in DEP_a$ aufgeführt, die die Relationen zwischen Schalt- und Ergebnismengen beschreiben. Die verschiedenen Bestandteile einer Spalte sind – analog zur Spaltenüberschrift – jeweils durch einen Schrägstrich voneinander getrennt.

Aktivität	**Eingangsschaltverhalten/ Schaltmengen**	**Ausgangsschaltverhalten/ Ergebnismengen/ Abhängigkeitsrelationen**
Anforderungs-analyse erstellen	complex / $\{17\}, \{3_1, 15_2, 17\}, \{3_1, 26_1\}$	dependent / $\{3_1, 30\}, \{3_1, 3_5\}, \{3_1, 3_2, 3_3, 3_4, 3_6\}$ / $\{17\} \times \{3_1, 30\}, \{3_1, 15_2, 17\} \times \{3_1, 3_5\}, \{3_1, 26_1\} \times \{\{3_1, 3_5\}, \{3_1, 3_2, 3_3, 3_4, 3_6\}\}$
Geschäftsprozess modellieren	det / $\{30\}, \{15_1\}$	all
technische Komponenten-anforderungen erstellen	complex / $\{3_2, 11_2\}, \{8_1, 9_1\}$	dependent / $\{8_1, 8_3\}, \{8_2\}$ / $\{3_2, 11_2\} \times \{8_1, 8_3\}, \{8_1, 9_1\} \times \{\{8_1, 8_3\}, \{8_2\}\}$
Review-durchführung	det / $\{3_5\}, \{8_3\}, \{11_4\}$	complex / $\{26_1, 26_2\}, \{9_1, 9_2\}$

Tabelle 3.4: Ein- und Ausgangsschaltverhalten der Aktivitäten aus der Beispielprozesslandschaft

Abbildung 3.20 zeigt den Ausschnitt der Prozesslandschaft aus Abbildung 3.13, deren Struktur durch die Konstruktion des kohärenten Petrinetzes und die Zuordnung der verschiedenen Schaltverhalten entstanden ist. Diese Schaltverhalten sind aus Gründen der Übersichtlichkeit in der Abbildung nicht explizit dargestellt, sondern nur in Tabelle 3.4 aufgeführt. Gleichzeitig ist zum besseren Überblick auch von allen bei der Zuordnung der Schaltverhalten nicht weiter berücksichtigten Aktivitäten abstrahiert worden. Damit ist neben den ursprünglich betrachteten drei Aktivitäten zusätzlich die Aktivität *Reviewdurchführung* in den Mittelpunkt der Beispielbetrachtung gerückt. Sie wird in Abschnitt 3.2.1.4 noch detaillierter diskutiert werden.



3_1,..., 3_6	Anforderungsdokument
8_1, 8_2 , 8_3	technische Komponentenanforderung
9_1, 9_2	Review technische Komponentenanforderung
11_2	Systemarchitektur
15_1,15_2, 15_3	Geschäftsprozessmodell
17	erste Anforderungen
26_1, 26_2	Review Anforderungsdokument
30	Anfrage nach Geschäftsprozessmodell

Abbildung 3.20: Kohärente Beispielprozesslandschaft mit zugeordneten Schaltverhalten

3.2.1.3 Anpassung der Verfeinerungskonzepte für die Modellierung der unteren Ebenen einer Prozesslandschaft

Im Folgenden werden Erweiterungen diskutiert, die eine Modellierung von Prozessen (als Ablauf sequentieller und paralleler Aktivitäten) und Informationen (als möglicherweise unterschiedliche Informationstypen) sowie ihre Beziehungen zueinander erlauben. Ergänzungen weiterer Informationen erfolgen unter anderem mittels Verfeinerungen und Erweiterungen der Landschaft, wie sie bereits in Abschnitt 3.1.1 eingeführt worden sind. Das Konzept der Verfeinerung muss jedoch für die unteren Ebenen einer Prozesslandschaft um zusätzliche Bedingungen ergänzt werden, um die Restriktionen für externe Schnittstellen zu erfüllen. Die Notwendigkeit dieser Anpassung ist bereits zu Beginn des Abschnitts 3.2.1 motiviert worden. Das Analyseziel, Synchronieaussagen in einer verteilten Prozesslandschaft machen zu können, hat dort die Unterscheidung in interne und externe Schnittstellen (vgl. Definition 3.2.4, Seite 96) zur Folge gehabt.

Konsistenzbedingung 3.2.34. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft mit $S = S_{intern} \cup S_{extern}$. Bei der Verfeinerung $PL'_{bottom} = (A', S', Z', AB')$ einer Prozesslandschaft PL_{bottom} bezüglich einer Aktivität a mit $S' = S'_{intern} \cup S'_{extern}$ müssen zusätzlich zu den in Definition 3.1.14 formulierten die folgenden Bedingungen für die externen Schnittstellen erfüllt sein:*

1. $\forall\, s \in {\bullet a} \cup a{\bullet} : \; s \in S'_{extern}$
2. $S_{extern} \subset S'_{extern}$

Bedingung 1. impliziert eine wichtige Vorbedingung für die Verfeinerung von Prozesslandschaften: Soll eine Prozesslandschaft PL_{bottom} bezüglich einer Aktivität a verfeinert werden, müssen zuvor alle Schnittstellen aus ihrem Vor- und Nachbereich als externe Schnittstellen deklariert werden. Dies kann eine Änderung der Modellierung derart erfordern, dass die Prozesslandschaft zur Erfüllung der Restriktionen für externe Schnittstellen vor dem Verfeinerungsschritt angepasst werden muss.

Beispiel:

Abbildung 3.21 zeigt die Aktivität *Release erstellen (CE)*, eine verfeinernde Aktivität des *Konfigurationsmanagement (CE)*. Anhand der fett umrandeten Schnittstellen *Releaseplanung* und *installiertes Softwaresystem* lässt sich leicht erkennen, dass noch weitere Aktivitäten auf den Vorbereich von *Release erstellen (CE)* zugreifen.

Soll die Aktivität *Release erstellen (CE)* detaillierter modelliert werden, muss ihr Vorbereich zunächst entsprechend dem ersten Teil der Konsistenzbedingung 3.2.34 angepasst werden. Dies geschieht mit Hilfe der (hier nicht abgebildeten) Dokumentensichten der betroffenen Schnittstellen, aus denen ersichtlich wird, dass insgesamt fünf verschiedene Aktivitäten auf die Schnittstelle *Releaseplanung* und insgesamt sechs verschiedene Aktivitäten auf die Schnittstelle *installiertes Softwaresystem* zugreifen.

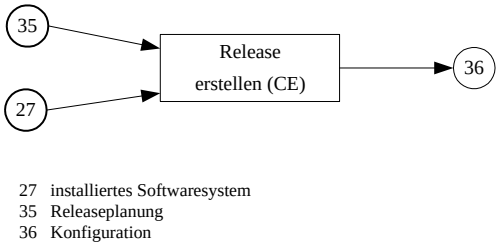


Abbildung 3.21: Aktivität *Release erstellen* zusammen mit ihrem Vor- und Nachbereich

Abbildung 3.22 zeigt das Ergebnis der Anpassungen gemäß der Restriktionen für externe Schnittstellen, die durch Duplizieren der beiden Schnittstellen *installiertes Softwaresystem* und *Releaseplanung* realisiert worden sind. Durch diese Form der Anpassung hat sich gleichzeitig die Ergebnismenge der beiden Aktivitäten *Releaseplanung (CE)* und *System installieren* vergrößert, die schreibend auf jeweils eine der betroffenen Schnittstellen zugreifen. Dies kann wiederum eine Anpassung ihres Schaltverhaltens notwendig machen. Insgesamt erfordern die Anpassungsschritte somit neben der Konsistenzsicherung für externe Schnittstellen auch eine Festlegung der Verfeinerungsreihenfolge von Landschaftselementen – konkret der Duplizierung beteiligter Schnittstellen vor der Verfeinerung zugreifender Aktivitäten – und eine mögliche Neufestlegung der Ausgangsschaltverhalten vorgelagerter Aktivitäten.

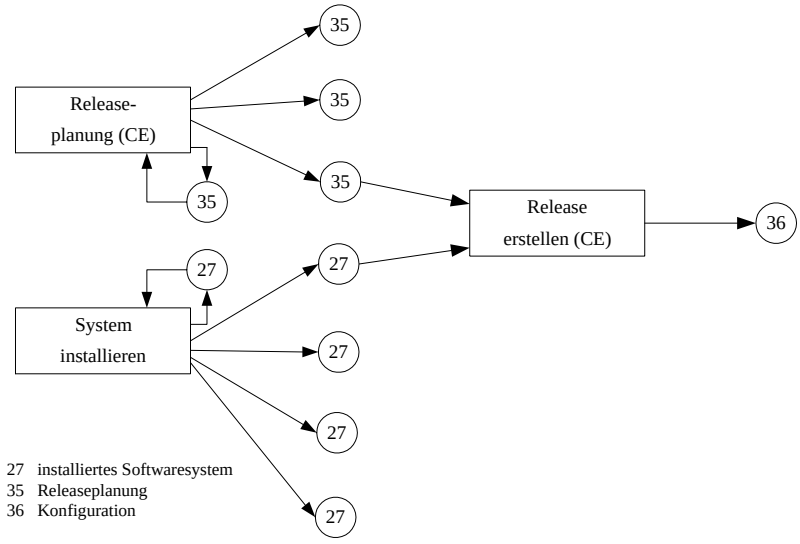


Abbildung 3.22: Anpassungen der Prozesslandschaft als Voraussetzung der Verfeinerung der Aktivität *Release erstellen (CE)*

Am gerade diskutierten Beispiel wird ein weiterer wesentlicher Unterschied zur traditionellen Verfeinerung von Petrinetzen deutlich: Beim Process Landscaping kann eine Verfeinerung Auswirkungen auf den Vor- und Nachbereich einer zu verfeinernden Aktivität und sogar auf das Schaltverhalten der im Ablauf vor- bzw. nachgelagerten Aktivitäten haben. Damit beschränkt sich die Verfeinerung im Gegensatz zur traditionellen Verfeinerung nicht mehr nur auf eine einzelne Transition, sondern auch auf ihre Umgebung. Die Einhaltung einer festen Reihenfolge von Verfeinerungsschritten wird dadurch um so wichtiger.

Konsistenzbedingung 3.2.35. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft mit $S = S_{intern} \cup S_{extern}$. Bei der Verfeinerung $PL'_{bottom} = (A', S', Z', AB')$ einer Prozesslandschaft PL_{bottom} bezüglich einer externen Schnittstelle s und $S' = S'_{intern} \cup S'_{extern}$ muss zusätzlich zu den in Definition 3.1.14 formulierten die folgende Bedingung für die internen und externen Schnittstellen erfüllt sein:*

$$S'_{intern} = S_{intern}$$

Die Bedingung fordert, dass die Menge der internen Schnittstellen in der verfeinernden Prozesslandschaft unverändert bleibt, eine externe Schnittstelle damit ausschließlich durch externe Schnittstellen verfeinert werden darf.

Für die Verfeinerung von Schnittstellen auf den unteren Ebenen einer Prozesslandschaft ist das zugehörige Verfeinerungskonzept damit im Vergleich zur traditionellen Vorgehensweise noch weiter eingeschränkt worden, da auf den unteren Ebenen nicht mehr alle, sondern nur noch die externen Schnittstellen verfeinert werden dürfen. Interne Schnittstellen müssen damit vor einer möglichen Verfeinerung zunächst an die Restriktionen der externen angepasst werden. Dies geschieht analog zum diskutierten Beispiel der Verfeinerung der Prozesslandschaft bezüglich einer Aktivität durch ihre Duplizierung.

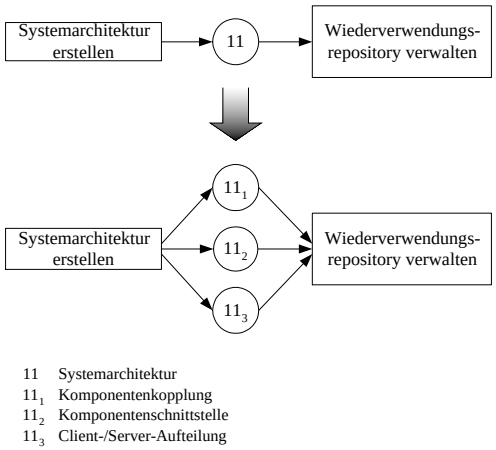


Abbildung 3.23: Verfeinerung einer externen Schnittstelle

Abbildung 3.23 zeigt ein Beispiel, bei dem die Schnittstelle *Systemarchitektur* durch die drei Schnittstellen *Komponentenkopplung*, *Komponentenschnittstellen* und *Client/Server-Aufteilung* verfeinert worden ist.

Für die Verfeinerung einer Prozesslandschaft bezüglich einer Aktivität ist das zugehörige Verfeinerungskonzept mit einer weiteren Konsistenzbedingung zu versehen:

Konsistenzbedingung 3.2.36. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft. Bei der Verfeinerung $PL'_{bottom} = (A', S', Z', AB')$ einer Prozesslandschaft PL_{bottom} bezüglich einer Aktivität a mit $A' = A \cup \{a_1, ..., a_n\}$ und $A \cap \{a_1, ..., a_n\} = \emptyset, n \geq 2$ muss zusätzlich zu den in Definition 3.1.14 formulierten die folgende Bedingung für die Menge Z' der Zugriffe erfüllt sein:*

$$\forall\, 1 \leq i \leq n : (\{a_i\} \times S) \cap Z' \neq \emptyset \wedge (S \times \{a_i\}) \cap Z' \neq \emptyset$$

Diese Konsistenzbedingung sichert den vollständigen Zusammenhang des der verfeinerten Prozesslandschaft PL'_{bottom} zugrundeliegenden Petrinetzes. Ein solcher Zusammenhang war für die oberen Ebenen aufgrund der fehlenden Ablaufinformationen nicht notwendig. Für die oberen Ebenen einer Prozesslandschaft war deshalb in Definition 3.1.14 lediglich gefordert, dass keine im Rahmen der Verfeinerung zusätzlich modellierte Schnittstelle isoliert ist. Für die verfeinernden Aktivitäten a_i war dies jedoch nicht gefordert.

Mit den diskutierten Konsistenzbedingungen sind die für die oberen Ebenen einer Prozesslandschaft definierten Konzepte der Verfeinerung von Aktivitäten und Schnittstellen auch für die unteren Ebenen einsetzbar. Damit ist gewährleistet, dass für die in Abschnitt 1.1 identifizierte Problemstellung – die mangelnde durchgängige Modellierungsunterstützung sowohl auf sehr abstraktem als auch auf sehr detailliertem Niveau – ein für alle Landschaftsebenen einheitliches Verfeinerungskonzept angewendet werden kann. Insgesamt ist somit für die in Abschnitt 2.2 diskutierten Modellierungsschritte des Process Landscaping eine umfassende formale Basis beschrieben.

3.2.1.4 Sonderfälle

Die in den vorangegangenen Abschnitten 3.2.1.1 und 3.2.1.2 durchgeführten Schritte reichen nicht immer aus, um den Kontext einer Prozesslandschaft in Bezug auf ihr Ablaufverhalten verständlich abzubilden. Wenn dieser Kontext eine konkrete Reihenfolge der Belegung verschiedener Ergebnismengen erfordert, kann bei einem nicht einfachen Ausgangsschaltverhalten die zufällige Auswahl einer von mehreren Ergebnismengen dazu führen, dass nicht alle modellierten Pfade durchlaufen werden können, obwohl dies aber im Rahmen der modellierten Ablaufsituation erforderlich wäre. Beispielsweise sollte in Abbildung 3.20 auf Seite 114 das Anforderungsdokument immer zuerst von der Aktivität *Reviewdurchführung* begutachtet werden, bevor es an die Aktivität *technische Komponentenanforderungen erstellen* weitergeleitet wird. Derartige Situationen sind dadurch entstanden, dass auf den oberen Ebenen einer Prozesslandschaft

keine Ablaufinformationen modelliert sind und ihr Hinzufügen auf den unteren Ebenen bislang ausschließlich durch syntaktische Regeln erfolgt ist. Dies ermöglicht dem Modellierer zwar einen einfachen und konsistenten Übergang zu den unteren Ebenen, kann aber teilweise auch zu der gerade beispielhaft beschriebenen Situation führen. Für die Handhabung derartiger Fälle werden im Folgenden spezielle Situationen diskutiert. Dies sind im Einzelnen:

- Sonderfall *Zusammengefasste Aktivität* (Eine Aktivität A empfängt – über ein komplexes Eingangsschaltverhalten – verschiedene Informationstypen und belegt in Abhängigkeit vom Eingangsschaltverhalten genau eine von mehreren disjunkten Ergebnismengen im Nachbereich, weist also ein abhängiges Ausgangsschaltverhalten auf.)
- Sonderfall *Zwei-Phasen-Aktivität* (In der ersten Phase wird nur eine Teilmenge aller Schnittstellen im Vorbereich einer Aktivität A als Schaltmenge verwendet, in der zweiten Phase müssen alle Schnittstellen im Vorbereich belegt sein. Da diese vollständige Belegung aber von A selbst in der ersten Phase über eine gesonderte Ergebnismenge initiiert wird, liegt auch in diesem Fall ein abhängiges Ausgangsschaltverhalten vor.)
- Sonderfall *Vier-Augen-Prinzip* (Auch hier liegt ein abhängiges Ausgangsschaltverhalten vor. Dieser Fall ähnelt der Zwei-Phasen-Aktivität insofern, als Aktivität A ebenfalls mehrfach ausgeführt werden kann, jedoch unter anderen Voraussetzungen für die vorhandenen Schnittstellen im Vorbereich.)

Die genannten Sonderfälle werden zunächst abstrakt diskutiert, anschließend in der Beispielprozesslandschaft (Abbildung 3.20, Seite 114) aufgezeigt und die entsprechend notwendigen Anpassungen zur Sicherstellung einer eindeutigen Darstellung des Ablaufverhaltens vorgenommen.

Zusammengefasste Aktivität:
Abbildung 3.24 stellt den Sonderfall einer zusammengefassten Aktivität dar. Die Aktivität A hat ein deterministisches Eingangs- und ein abhängiges Ausgangsschaltverhalten, wobei das deterministische Eingangsschaltverhalten – wie bereits erwähnt – einen Spezialfall des komplexen Eingangsschaltverhaltens darstellt. A belegt nach dem Schalten in Abhängigkeit der Schaltmenge im Vorbereich jeweils eine andere Ergebnismenge in ihrem Nachbereich. Die bei der Ausführung der Aktivität verwendeten und erzeugten Informationstypen gehören also paarweise zueinander. Des Weiteren gilt, dass die Schaltmengen paarweise disjunkt sind. Formal:

- $f_{source}(A) = det$ mit
  $S_{source,A-1} = \{a\}, S_{source,A-2} = \{b\}, S_{source,A-3} = \{c\}$
- $f_{dest}(A) = dependent$ mit
  $S_{dest,A-1} = \{d\}, S_{dest,A-2} = \{e\}, S_{dest,A-3} = \{f\}$

- $DEP_A = (\{a\} \times \{d\}, \{b\} \times \{e\}, \{c\} \times \{f\})$
- $|(H_{source,A})| = |(H_{dest,A})| > 1$

Die Forderung, dass die Anzahl der Schalt- und Ergebnismengen größer eins sein muss, ergibt sich aus dem abhängigen Ausgangsschaltverhalten der Aktivität A (vgl. Definition 3.2.26, Seite 111). In Abhängigkeit der Anzahl vorliegender Ergebnismengen kann jede Aktivität A durch eine entsprechende Menge von Teilaktivitäten A_i verfeinert werden.

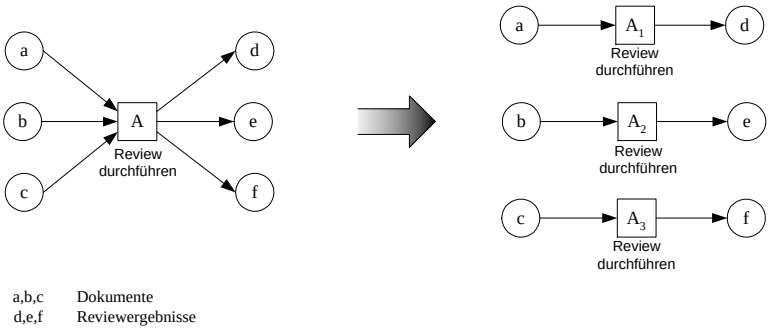


Abbildung 3.24: Zusammengefasste Aktivität

Durch eine Verfeinerung von A entsteht eine semantisch sinnvolle Verfeinerung, wie sie rechts in Abbildung 3.24 dargestellt ist. Die Aktivität A wurde hier durch die Aktivitäten A_1, A_2 und A_3 verfeinert. Diesen wird jeweils das einfache Eingangs- und Ausgangsschaltverhalten zugeordnet. Als Beispiel für eine konkrete Situation dieses Sonderfalls ist in der Abbildung eine Reviewtätigkeit dargestellt, die in Abhängigkeit vom Vorhandensein eines Dokumentes a, b oder c eines der Reviewergebnisse d, e oder f produziert.

Zwei-Phasen-Aktivität:

Eine Zwei-Phasen-Aktivität zeichnet sich dadurch aus, dass die vollständige Durchführung der ihr zugewiesenen Operationen von dem Ergebnis einer zweiten Aktivität abhängt. Die linke Seite der Abbildung 3.25 zeigt eine solche Situation.

Die Aktivität A hat ein komplexes Eingangs- und ein abhängiges Ausgangsschaltverhalten, ihre Ergebnismengen sind disjunkt. Es gilt:

- $f_{source}(A) = complex$ mit
  $S_{source,A-1} = \{a\}, S_{source,A-2} = \{a, c\}$
- $f_{dest}(A) = dependent$ mit
  $S_{dest,A-1} = \{b\}, S_{dest,A-2} = \{d\}$
- $DEP_A = (\{a\} \times \{b\}, \{a, c\} \times \{d\})$

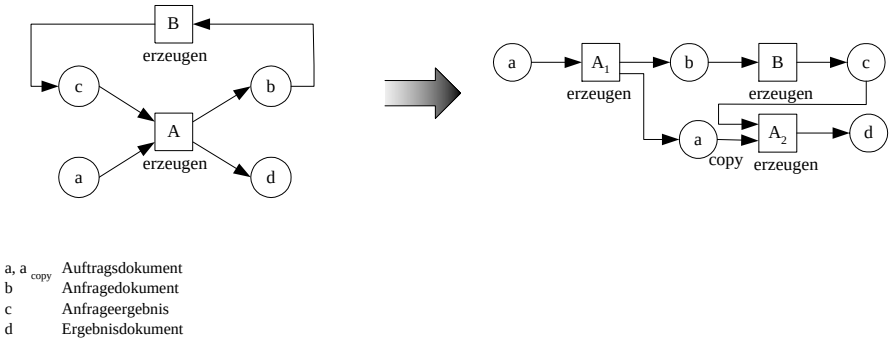


Abbildung 3.25: Zwei-Phasen-Aktivität

In einer ersten Phase schaltet A aufgrund eines an der Schnittstelle a vorliegenden (gleichnamigen) Informationsobjektes. Die Erzeugung des Informationsobjektes d ist jedoch abhängig vom Vorliegen des Informationsobjektes c, welches durch die Aktivität B erzeugt wird. Dazu erzeugt A zunächst das Informationsobjekt b, Aktivität B kann schalten und c erzeugen. In einer zweiten Phase kann A schließlich d erzeugen. Um die zeitliche Abhängigkeit der beiden Durchläufe transparent zu machen, wird Aktivität A durch die Aktivitäten A_1 und A_2 verfeinert.

Damit ist analog zur zusammengefassten Aktivität auch bei der Zwei-Phasen-Aktivität eine Abhängigkeit zwischen Schalt- und Ergebnismengen identifiziert. Eine analoge Verfeinerung ist jedoch nicht direkt möglich, da ein in der Schnittstelle a vorliegender Informationstyp für den Durchlauf beider Phasen benötigt wird. Das zugrundeliegende Petrinetz wird daher um eine interne Schnittstelle a_{copy} – eine lokale Kopie der externen Schnittstelle a – erweitert, auf die die Aktivität A sowohl lesend als auch schreibend zugreift. Nach dieser Erweiterung kann eine Verfeinerung der Aktivität A analog zur zusammengefassten Aktivität durchgeführt werden.

Zur Identifikation einer Zwei-Phasen-Aktivität A reicht damit das Vorhandensein eines abhängigen Ausgangsschaltverhaltens nicht aus. Es muss auch eine Wechselbeziehung zu einer zweiten Aktivität B vorhanden sein. Für diese muss eine Schaltmenge $S_{source,B}$ identisch mit einer Ergebnismenge $S_{dest,A}$ der Aktivität A sein. Umgekehrt muss eine Ergebnismenge $S_{dest,B}$ der Aktivität B identisch mit einer Schaltmenge $S_{source,A}$ der Aktivität A sein oder zumindest $S_{source,A}$ als Teilmenge enthalten.

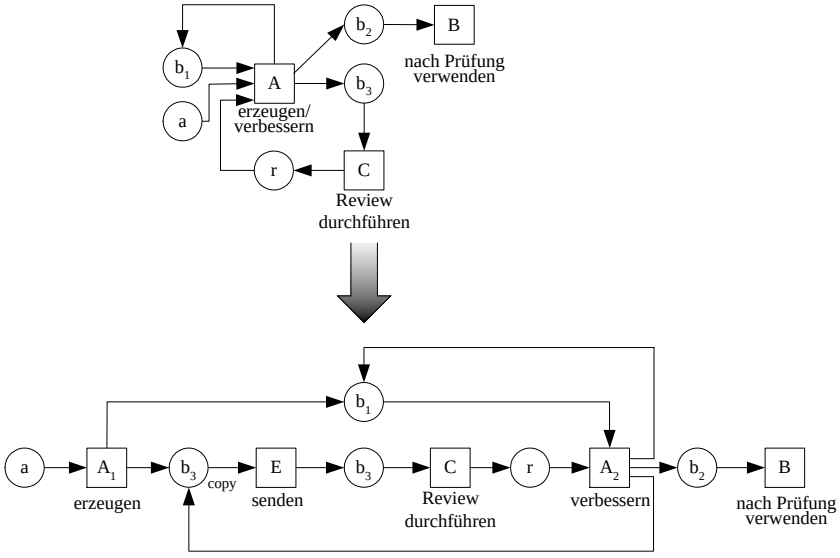
Definition 3.2.37. *Sei $PN = (A_{PN}, S, Z_{PN})$ ein kohärentes Petrinetz. Eine Aktivität $a \in A_{PN}$ ist eine **Zwei-Phasen-Aktivität** :⇔*

1. $f_{source}(a) = complex$
2. $f_{dest}(a) = dependent$
3. $\exists\, S_{dest,a} \in (H_{dest,a}), b \in A, S_{source,b} \in (H_{source,b}) : S_{dest,a} \supseteq S_{source,b}$
4. $\exists\, S_{source,a} \in (H_{source,a}), b \in A, S_{dest,b} \in (H_{dest,b}) : S_{source,a} \supseteq S_{dest,b}$

Die rechte Seite der Abbildung 3.25 zeigt die entsprechende Anpassung bzw. Verfeinerung des Sonderfalls *Zwei-Phasen-Aktivität*: Aktivität A_1 erzeugt das Informationsobjekt b sowie eine Kopie des Informationsobjektes a. Für Letzteres ist das zugrundeliegende Petrinetz um eine Schnittstelle a_{copy} und zwei Zugriffe (A_1, a_{copy}) und (a_{copy}, A_2) erweitert worden, über die die Aktivitäten A_1 und A_2 das Informationsobjekt a austauschen. Liegt b in der gleichnamigen Schnittstelle vor, kann Aktivität B das Objekt c erzeugen. Die Schnittstellen c und a_{copy} bilden schließlich die Schaltmenge für Aktivität A_2, so dass bei deren Belegung A_2 schalten und das erzeugte Informationsobjekt d auf der entsprechenden Schnittstelle abgelegt werden kann. Den Aktivitäten A_1 und A_2 ist jeweils ein einfaches Ein- und Ausgangsschaltverhalten zugeordnet.

Vier-Augen-Prinzip:

Der obere Teil der Abbildung 3.26 zeigt den Sonderfall des Vier-Augen-Prinzips. Bei diesem erzeugt die Aktivität A das Informationsobjekt b und speichert es in den Schnittstellen b_1 und b_2 bzw. b_3. Bevor es von Aktivität B verwendet werden kann, muss es von Aktivität C geprüft werden. In Abhängigkeit des Prüfungsergebnisses (Informationsobjekt c) überarbeitet Aktivität A das Informationsobjekt b noch einmal und stellt es entweder Aktivität C für eine erneute Prüfung zur Verfügung oder gibt es Aktivität B zur weiteren Verwendung weiter.



a	Startdokument
$b_1, b_2, b_3, b_{3,\,copy}$	erzeugtes Dokument
r	Reviewergebnis

Abbildung 3.26: Vier-Augen-Prinzip

Für das Schaltverhalten der Aktivität A gilt:

- $f_{source}(A) = complex$ mit
  $S_{source,A-1} = \{a\}, S_{source,A-2} = \{b_1, r\}$
- $f_{dest}(A) = dependent$ mit $S_{dest,A-1} = \{b_1, b_3\}, S_{dest,A-2} = \{b_2\}$
- $DEP_A = (\{a\} \times \{b_1, b_3\}, \{b_1, r\} \times \big\{\{b_1, b_3\}, \{b_2\}\big\})$

Damit ist auch für diesen Sonderfall eine Abhängigkeit zwischen Schalt- und Ergebnismengen gegeben, und eine entsprechende Verfeinerung der Aktivität A durch zwei Teilaktivitäten (aufgrund zweier vorliegender Schaltmengen) kann vorgenommen werden.

Eine kausale – und damit auch zeitliche – Abhängigkeit für das Schalten der verschiedenen Aktivitäten liegt ebenfalls vor, da Aktivität C mindestens einmal geschaltet haben muss, bevor Aktivität B schalten darf. Andernfalls könnten Dokumente ungeprüft weiterverwendet werden. Diese zeitliche Abhängigkeit ist im vorliegenden Petrinetz im oberen Teil der Abbildung 3.26 nicht erkennbar. Sie wird erst durch eine verfeinerte Modellierung der Ablaufsituation deutlich, die im unteren Teil der Abbildung 3.26 dargestellt ist: Aktivität A ist jetzt durch die Aktivitäten A_1, E und A_2 detaillierter dargestellt, wobei A_1 und E ein einfaches Eingangs- und Ausgangsschaltverhalten haben, A_2 dagegen zwar ein einfaches Eingangs- aber ein komplexes Ausgangsschaltverhalten. Die Aktivität E ist eine Senderaktivität (vgl. Abbildung 3.17, Seite 107), die im Vorbereich der externen Schnittstelle b_3 zwischengefügt wurde, um die Restriktionen für externe Schnittstellen nicht zu verletzen. Sie wird über eine interne Schnittstelle $b_{3,\,copy}$ aktiviert und sendet von Aktivität A_1 erzeugte Informationstypen an die Aktivität C.

Die Anwendungen der beschriebenen Sonderfälle sind Verfeinerungsschritte, die sich aus konkreten Ablaufsituationen ergeben. Sie sind bereits an der Struktur des zugrundeliegenden Petrinetzes erkennbar. Allen ist eine Abhängigkeit zwischen Schalt- und Ergebnismengen gemeinsam (abhängiges Ausgangsschaltverhalten), die sich im Sonderfall der zusammengesetzten Aktivität durch eine 1:1-Beziehung ausdrückt, ansonsten durch eine n:m-Beziehung. Wechselbeziehungen zu anderen Aktivitäten werden, falls vorhanden, nach der Anwendung der Sonderfälle durch die Anordnung der verfeinernden Aktivitäten deutlich. Eine Zwei-Phasen-Aktivität weist zusätzlich immer Zugriffe zu Schalt- und Ergebnismengen einer zweiten Aktivität auf. Die tatsächliche Anwendung der Sonderfälle muss jedoch dem Modellierer überlassen werden, da Verfeinerungen innerhalb einer Prozesslandschaft vom jeweiligen Modellierungsziel abhängig sind.

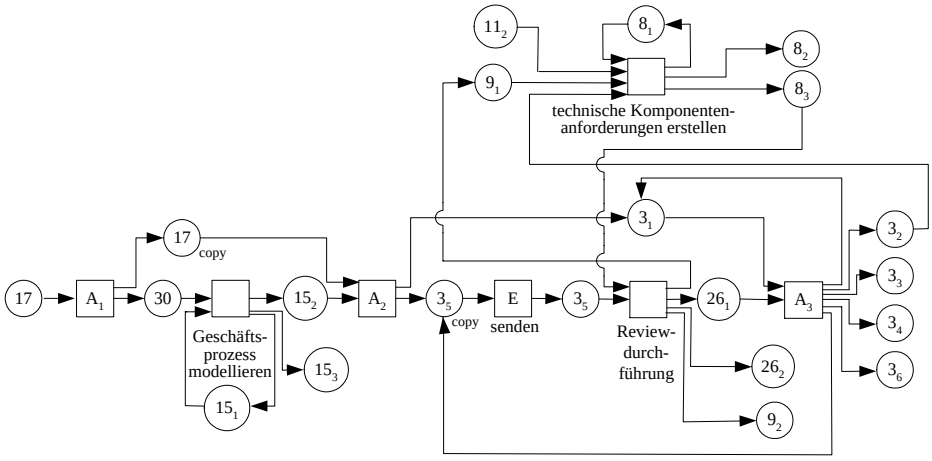
In der diskutierten Beispiellandschaft können nach der Zuweisung der verschiedenen Schaltverhalten (vgl. Tabelle 3.4 und Abbildung 3.20, Seite 113) alle drei Sonderfälle identifiziert werden:

- Die Aktivität *Anforderungsanalyse erstellen* entspricht einer Zwei-Phasen-Aktivität, da sie für die Erstellung eines Anforderungsdokuments zunächst das Ge-

schäftsprozessmodell anfordert. Da sie ihr Ergebnis – das Anforderungsdokument – einem Review unterzieht und in Abhängigkeit vom Reviewergebnis gegebenenfalls Verbesserungen einarbeiten muss, liegt eine n:m-Beziehung zwischen ihren Schalt- und Ergebnismengen vor. Bei drei Ergebnismengen kann die Aktivität entsprechend durch drei verfeinernde Aktivitäten dargestellt werden.

- Die Aktivität *Reviewdurchführung* ist eine zusammengefasste Aktivität, da sie in Abhängigkeit des eingehenden Dokumentes jeweils ein korrespondierendes Reviewergebnis produziert. Diese Abhängigkeit lässt sich über eine 1:1-Beziehung ihrer Schalt- und Ergebnismengen formulieren. Mit zwei identifizierten Ergebnismengen kann die Aktivität durch zwei verfeinernde Aktivitäten dargestellt werden.

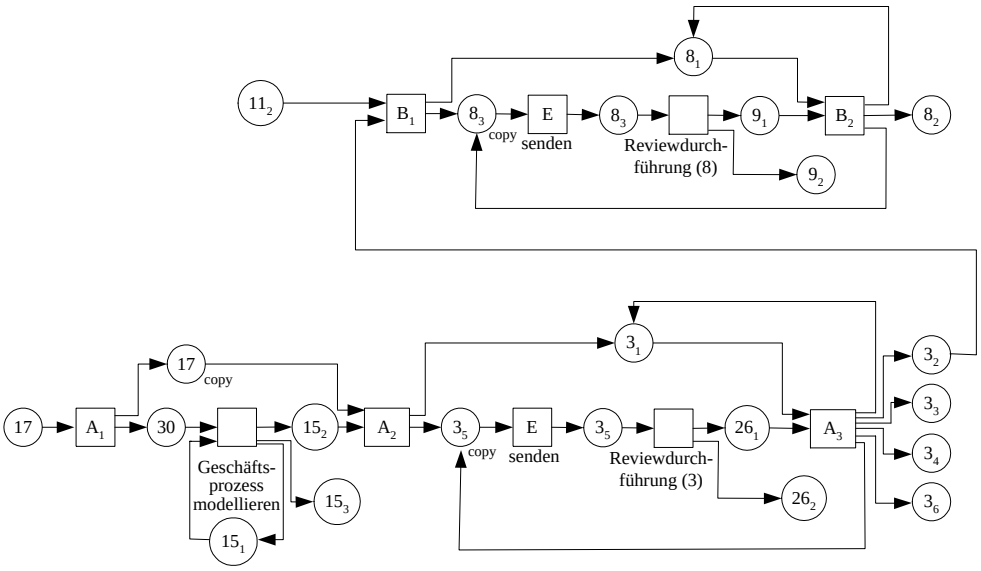
- Die Aktivität *technische Komponentenanforderungen erstellen* repräsentiert den Sonderfall des Vier-Augen-Prinzips, da ihr Ergebnis einem Review unterzogen wird und sie eventuell erneut durchlaufen werden muss. Die Relationen zwischen ihren Schalt- und Ergebnismengen lassen sich als n:m-Beziehung formulieren. Auch in diesem Fall liegen zwei Ergebnismengen vor. Entsprechend kann die Aktivität durch zwei verfeinernde Aktivitäten dargestellt werden.



3_1,..., 3_6, $3_{5,copy}$	Anforderungsdokument	A_1, A_2, A_3	Anforderungsanalyse erstellen
8_1,..., 8_3	technische Komponentenanforderung		
9_1, 9_2	Review technische Komponentenanforderung		
11_2	Systemarchitektur		
15_1, 15_2, 15_3	Geschäftsprozessmodell		
17, 17_{copy}	erste Anforderungen		
26_1, 26_2	Review Anforderungsdokument		
30	Anfrage nach Geschäftsprozessmodell		

Abbildung 3.27: Beispielprozesslandschaft nach Anpassungen bezüglich des Sonderfalls *Zwei-Phasen- Aktivität*

In Abbildung 3.27 sind zunächst die Anpassungen an der Aktivität *Anforderungsanalyse erstellen* dargestellt. Sie ist jetzt durch die drei Aktivitäten A_1, A_2 und A_3 verfeinert. Entsprechend des Sonderfalls der Zwei-Phasen-Aktivität wurde das zugrundeliegende Petrinetz zudem durch eine Kopie der Schnittstelle *erste Anforderungen* inklusive zugehöriger Zugriffe erweitert. Da die Schnittstelle 3_5 eine externe Schnittstelle ist, wurde sie entsprechend ihrer Restriktionen als interne kopiert. Zusätzlich wurde eine Senderaktivität E eingefügt. So werden die temporalen Abhängigkeiten der drei Teilaktivitäten transparent und die Ablaufinformationen verdeutlicht. Die verfeinernden Aktivitäten A_1, A_2 und A_3 sowie die Senderaktivität E haben jeweils ein einfaches Eingangsschaltverhalten.



3_1,..., 3_6, $3_{5,copy}$	Anforderungsdokument	A_1, A_2, A_3	Anforderungsanalyse erstellen
8_1,..., 8_3, $8_{3,copy}$	technische Komponentenanforderung	B_1, B_2	technische Komponentenanforderungen erstellen
9_1, 9_2	Review technische Komponentenanforderung		
11_2	Systemarchitektur		
15_1, 15_2, 15_3	Geschäftsprozessmodell		
17, 17_{copy}	erste Anforderungen		
26_1, 26_2	Review Anforderungsdokument		
30	Anfrage nach Geschäftsprozessmodell		

Abbildung 3.28: Beispielprozesslandschaft nach Anpassungen bezüglich der Sonderfälle *Zusammengefasste Aktivität* und *Vier-Augen-Prinzip*

Abbildung 3.28 zeigt die Anpassungen der Beispielprozesslandschaft, die sich aus dem Sonderfall *Zusammengefasste Aktivität* für die Aktivität *Reviewdurchführung* ergeben sowie die Anpassungen, die sich aus dem Sonderfall *Vier-Augen-Prinzip* für die Aktivität *technische Komponentenanforderung erstellen* ergeben. Die Aktivität *Reviewdurchführung* ist durch die beiden Aktivitäten *Reviewdurchführung(3)* und *Reviewdurchführung(8)* verfeinert worden, die Aktivität *technische Komponentenanforderun-*

gen erstellen entsprechend durch die Aktivitäten B_1 und B_2. Da die Schnittstelle 8_3 eine externe Schnittstelle ist, wurde das zugrundeliegende Petrinetz ebenfalls entsprechend ihrer Restriktionen angepasst.

Mit der Betrachtung verschiedener Sonderfälle ist die Diskussion bezüglich der Ergänzung einer Prozesslandschaft um Ablaufinformationen abgeschlossen, wobei für die Erweiterung des zugrundeliegenden Petrinetzes zu einem kohärenten Netz nur die nicht weiter verfeinerten Aktivitäten $a \in leaves$ berücksichtigt wurden. Bislang ist damit nur eine konkrete Repräsentation einer hierarchisch strukturierten Prozesslandschaft diskutiert worden. Diese Repräsentation zeigt die Prozesslandschaft als ein kohärentes, teilweise verfeinertes und nicht hierarchisches Petrinetz, bestehend aus allen Blättern des Aktivitätenbaumes AB.

Damit ist eine Prozesslandschaft PL_{bottom} immer eine konkrete Repräsentation aus einer Familie von Prozesslandschaftsrepräsentationen, die einen gemeinsamen Aktivitätenbaum AB haben. Es existieren andere mögliche Repräsentationen der gleichen Prozesslandschaft, bei denen Teilnetze zu einer einzigen Aktivität $a \in ref$ zusammengefasst sind und so von Verfeinerungen abstrahiert wird. Eine solche Vergröberung und die daraus resultierende neue Repräsentation einer Prozesslandschaft kann durch eine Abbildung erstellt werden. Bevor diese Abbildung definiert werden kann, sind noch einige Vorüberlegungen notwendig.

Definition 3.2.38. *Sei $a_{fold} \in ref$ ein innerer Knoten des Aktivitätenbaumes AB der dritten Ausbaustufe einer Prozesslandschaft PL_{bottom}. Die Menge $\mathbf{empty_{a_{fold}}}$ enthält alle Nachfolger – auch die nicht direkten – der Aktivität a_{fold}:*

$$empty_{a_{fold}} := succ^{+}(a_{fold})$$

Die Menge $\mathbf{empty_{s_{intern}}}$ enthält alle Schnittstellen $s \in S$, zu denen Zugriffe $z \in Z$ von Elementen der Menge $empty_{a_{fold}}$ existieren:

$$empty_{s_{intern}} := \{s \in S \mid \exists\, (s,a),(a',s) \in Z : a, a' \in empty_{a_{fold}}\}$$

Die Menge $\mathbf{empty_z}$ enthält alle Zugriffe $z \in Z$ von Elementen aus $empty_{a_{fold}}$ zu Elementen aus $empty_{s_{intern}}$:

$$\begin{aligned} empty_z :=& \{(a,s) \in Z \mid a \in empty_{a_{fold}} \wedge s \in empty_{s_{intern}}\} \cup \\ & \{(s,a) \in Z \mid s \in empty_{s_{intern}} \wedge a \in empty_{a_{fold}}\} \end{aligned}$$

Die Menge $\mathbf{replace_z}$ enthält alle Zugriffe $z \in Z$ von Elementen aus $empty_{a_{fold}}$ zu Elementen aus $S \backslash empty_{s_{intern}}$:

$$\begin{aligned} replace_z :=& \{(a,s) \in Z \mid a \in empty_{a_{fold}} \wedge s \in S \backslash empty_{s_{intern}}\} \cup \\ & \{(s,a) \in Z \mid s \in S \backslash empty_{s_{intern}} \wedge a \in empty_{a_{fold}}\} \end{aligned}$$

Die Menge $\mathbf{substitution_z}$ enthält Zugriffe $z \in Z$ von a_{fold} zu Elementen aus $S \backslash empty_{s_{intern}}$:

$$\begin{aligned} substitution_z :=& \{(a_{fold}, s) \mid (a,s) \in replace_z\} \cup \\ & \{(s, a_{fold}) \mid (s,a) \in replace_z\} \end{aligned}$$

Notation 3.2.39. *Ein Element $a_{fold} \in ref$ wird als **Repräsentationsaktivität** bezeichnet.*

Damit sind alle für eine Definition der Abbildung $fold$ notwendigen Vorarbeiten geleistet.

Definition 3.2.40. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft. Die Abbildung* **fold** *liefert weitere Petrinetz-Repräsentationen der Prozesslandschaft PL_{bottom}, bei der genau ein Teilbaum aus AB nicht detailliert dargestellt ist, sondern durch eine Repräsentationsaktivität a_{fold} repräsentiert ist:*

$$fold : (A_{PN}, S, Z_{PN}, a_{fold}) \rightarrow (A'_{PN}, S', Z'_{PN}, a_{fold}) \qquad \textit{wobei}$$

$$\begin{aligned} A'_{PN} &:= A_{PN} \backslash empty_{a_{fold}} \\ S' &:= S \backslash empty_{s_{intern}} \\ Z'_{PN} &:= \big(Z_{PN} \backslash (empty_z \cup replace_z)\big) \cup substitution_z \end{aligned}$$

Die Menge $empty_{a_{fold}}$ enthält dabei alle Aktivitäten $a \in A_{PN}$, die nach Anwendung der Abbildung $fold$ nicht mehr explizit dargestellt sind, sondern durch die Aktivität a_{fold} repräsentiert werden.
Die Menge $empty_{s_{intern}}$ enthält alle Schnittstellen $s \in S$, die nach Anwendung der Abbildung $fold$ nicht mehr explizit dargestellt sind.
Die Menge $empty_z$ enthält alle Zugriffe $z \in Z_{PN}$, die nach Anwendung der Abbildung $fold$ nicht mehr explizit dargestellt sind.
Die Menge $replace_z$ enthält alle Zugriffe $z \in Z_{PN}$, die durch die Anwendung der Abbildung $fold$ durch Zugriffe (s, a_{fold}) oder (a_{fold}, s) ersetzt werden.
Die Menge $substitution_z$ enthält Zugriffe z, die nach Anwendung der Abbildung $fold$ auf die Menge $replace_z$ neu in der Petrinetz-Repräsentation entstanden sind.

Bemerkung 3.2.41. *Mehrfachanwendungen der Abbildung $fold$ führen zu Repräsentationen einer Prozesslandschaft, bei denen mehrere Teilbäume aus AB durch verschiedene Repräsentationsaktivitäten a_{fold_i} dargestellt sind. Vor einer Mehrfachanwendung von $fold$ muss geprüft werden, ob die verschiedenen a_{fold} in der transitiven Hülle eines jeweils anderen a'_{fold} liegen. Ist dies für zwei a_{fold} und a'_{fold} der Fall, so gilt entweder $a_{fold} \in succ^+(a'_{fold})$ oder $a'_{fold} \in succ^+(a_{fold})$ (vgl. Bemerkung 3.1.5). Im ersten Fall kann daher auf die Anwendung von $fold$ bezüglich der Repräsentationsaktivität a_{fold}, andernfalls auf die Anwendung von $fold$ bezüglich a'_{fold} verzichtet werden.*

Eine Prozesslandschaft PL_{bottom} kann so mit Hilfe der Abbildung $fold$ in unterschiedlichen Detaillierungsgraden als Prozesslandschaft PL'_{bottom} formuliert werden.

3.2.2 Erweiterungen zur Analyse dynamischer Kommunikationseigenschaften

Dieser Abschnitt führt zusätzliche Erweiterungen des einer Prozesslandschaft zugrundeliegenden Petrinetzes ein. Diese unterstützen insbesondere die Analyse dynamischer

Kommunikationseigenschaften in einer verteilten Prozesslandschaft. Zuvor wird jedoch die Umstrukturierung einer gegebenen Prozesslandschaft diskutiert, deren Ergebnis sich auf den Verteilungsaspekt der Aktivitäten auf verschiedene Standorte konzentriert und so die Betrachtung spezieller Kommunikationseigenschaften erleichtert.

3.2.2.1 Umstrukturierung einer Prozesslandschaft in eine lokale Sicht

Bislang wurde bei einer Prozesslandschaft immer davon ausgegangen, dass ihre Verfeinerung nach logischen Gesichtspunkten durchgeführt wird, was bedeutet, dass komplexe Aktivitäten durch eine Menge einfacherer Aktivitäten verfeinert werden und auch die Schnittstellen zwischen ihnen den Inhalt der auszutauschenden Informationstypen immer detaillierter darstellen. Die Sicht auf eine so entstandene hierarchische Prozesslandschaft spiegelt vor allem die temporalen und kausalen Zusammenhänge wider. Verteilungsaspekte sind durch die Abbildung $local$ berücksichtigt, sie sind jedoch in der grafischen Repräsentation einer Prozesslandschaft nicht erkennbar, da die Lokalitätsattribute von Aktivitäten hier bislang nicht berücksichtigt sind.

Notation 3.2.42. *Eine nach logischen Gesichtspunkten strukturierte Prozesslandschaft PL wird im Folgenden kurz als $\mathbf{PL_{logic}}$ bzw. als **logische Sicht** einer Prozesslandschaft bezeichnet.*

Um die Analyse einer über mehrere Standorte verteilten Prozesslandschaft durch eine geeignete grafische Repräsentation zu unterstützen, kann die logische Sicht derart umstrukturiert werden, dass sich für die Aktivitäten eine neue Anordnung ergibt. Diese spiegelt dann die lokale Verteilung aller Aktivitäten in einer Prozesslandschaft wider. Ihre Anordnungskriterien resultieren aus den verschiedenen Standorten und deren Strukturen.

Notation 3.2.43. *Eine nach lokalen Gesichtspunkten strukturierte Prozesslandschaft PL wird im Folgenden kurz als $\mathbf{PL_{local}}$ bzw. als **lokale Sicht** einer Prozesslandschaft bezeichnet.*

In Abschnitt 2.3.1 wurde bereits am Beispiel eines Puzzles abstrakt dargestellt, welche Auswirkungen eine Umstrukturierung von der logischen in die lokale Sicht auf die grafische Repräsentation der Prozesslandschaft haben kann (vgl. Abbildung 2.6, Seite 35). Im folgenden Abschnitt wird auf die formalen Grundlagen der Umstrukturierung eingegangen.

Wird eine Prozesslandschaft PL_{logic} nach lokalen Gesichtspunkten in eine Prozesslandschaft PL_{local} umstrukturiert, werden Aktivitäten, die in PL_{logic} nur einmal modelliert sind, in PL_{local} immer dann mehrfach dargestellt, wenn sie an mehreren Orten durchgeführt werden, die Kardinalität ihres Lokalitätsattributs also größer eins ist. Dies wird insbesondere am Beispiel der Prozesslandschaft zur komponentenbasierten Softwareentwicklung deutlich, wenn mehrere Komponenten an unterschiedlichen Orten parallel entwickelt werden. Um die so entstehenden zusätzlichen Aktivitäten der Prozesslandschaft PL_{local} eindeutig auf ihren Ursprung in der logischen Sicht zurückführen zu können, werden sie in PL_{local} als Paar formuliert:

Definition 3.2.44. *Sei $PL_{logic} = (A_{logic}, S_{logic}, Z_{logic}, AB_{logic})$ die dritte Ausbaustufe einer nach logischen Gesichtspunkten modellierten Prozesslandschaft und $O \neq \emptyset$ eine gültige Ortemenge, deren Elemente mit Hilfe der Abbildung $local$ den Aktivitäten der Prozesslandschaft zugeordnet sind.*
Sei weiterhin $PL_{local} = (A_{local}, S_{local}, Z_{local}, AB_{local})$ die dritte Ausbaustufe einer nach lokalen Gesichtspunkten umstrukturierte Prozesslandschaft, die aus der Umstrukturierung von PL_{logic} entstanden ist. Für die Aktivitäten aus A_{logic} gilt:

$$\forall\, a \in A_{logic} : A_{local,a} = \{a\} \times local(a)$$

Die Menge A_{local} lässt sich definieren als

$$A_{local} := \bigcup_{a \in A_{logic}} A_{local,a}$$

Mit der Formulierung der Aktivitäten aus A_{local} ist immer eindeutig erkennbar, welche Aktivität aus A_{logic} mit welchen Aktivitäten aus A_{local} korrespondiert. Durch das Kreuzprodukt einer Aktivität aus A_{logic} mit ihren Lokationsattributen ist jedoch nicht nur diese eindeutige Zuordnung gewährleistet, es wird auch gleichzeitig die Anzahl der in einer Prozesslandschaft PL_{local} korrespondierenden Aktivitäten festgelegt.

Die Schnittstellen der in PL_{local} mehrfach vorhandenen Aktivitäten zu anderen werden ebenfalls entsprechend mehrfach dargestellt, um spätere Kommunikationsanalysen zwischen verschiedenen Standorten zu ermöglichen. Die so entstehenden zusätzlichen Elemente der Prozesslandschaft PL_{local} müssen verschiedenen Bedingungen genügen. Zuvor werden jedoch alle Schnittstellen $s \in S_{logic}$, über die zwei Aktivitäten unterschiedlicher Standorte Informationen austauschen, zu externen Schnittstellen (vgl. Definition 3.2.4 auf Seite 96) umformuliert. Dies erleichtert nicht nur die Umstrukturierung in eine lokale Sicht, sondern unterstützt vor allem die angesprochene Kommunikationsanalyse.

Konsistenzbedingung 3.2.45. *Sei $PL_{logic} = (A_{logic}, S_{logic}, Z_{logic}, AB_{logic})$ die logische Sicht auf die dritte Ausbaustufe einer Prozesslandschaft und $S_{extern} \subseteq S_{logic}$ die Menge der externen Schnittstellen in PL_{logic}.*

$$\forall\, s \in S_{logic} : \bigcup_{a \in \bullet s} local(\bullet s) \neq \bigcup_{a \in s\bullet} local(s\bullet) \Rightarrow s \in S_{extern}$$

Durch diese Bedingung werden alle Schnittstellen, die zwischen zwei oder mehr Orten liegen, zu externen Schnittstellen. Sie müssen gegebenenfalls gemäß der in Definition 3.2.4 auf Seite 96 für sie formulierten Restriktionen angepasst werden. Die dazu erforderlichen Schritte sind bereits durch die Abbildungen 3.15 (Seite 106) und 3.17 (Seite 107) erläutert worden.

Definition 3.2.46. *Sei $PL_{logic} = (A_{logic}, S_{logic}, Z_{logic}, AB_{logic})$ die dritte Ausbaustufe einer nach logischen Gesichtspunkten modellierten Prozesslandschaft und $O \neq \emptyset$*

eine gültige Ortemenge, deren Elemente mit Hilfe der Abbildung $local$ den Aktivitäten der Prozesslandschaft zugeordnet sind. Für die lokale Sicht $PL_{local} = (A_{local}, S_{local}, Z_{local}, AB_{local})$ der Prozesslandschaft können die Mengen S_{local}, Z_{local} und AB_{local} wie folgt definiert werden:
Für alle $s_i \in S_{logic}$ mit $S_{logic} = \{s_1, ..., s_n\}, n \in \mathbb{N}$ und $\bullet s = \{a_i\}, s\bullet = \{a_i'\}$ mit $a_i, a_i' \in A_{logic}, a_i \neq a_i'$ wird eine Teilmenge

$$T_i \subseteq A_{local,a_i} \times A_{local,a_i'}$$

als diejenige Menge aller Aktivitätenpaare der lokalen Sicht gewählt, die miteinander kommunizieren. Dann gelte für die Menge der Schnittstellen $S_i = \{s_{i,1}, ..., s_{i,|T_i|}\}$, über die die Aktivitätenpaare kommunizieren, dass

$$\{s_{i,1}, ..., s_{i,|T_i|}\} \cap S_{logic} = \emptyset$$
$$\forall\, 1 \leq i, j \leq n, i \neq j : S_i \cap S_j = \emptyset$$

Somit kann S_{local} definiert werden als

$$S_{local} := \big(S_{logic} \backslash S_{extern}\big) \cup \{s_{i,j} \mid 1 \leq i \leq n, 1 \leq k \leq |T_i|\}$$

Für Z_{local} gilt

$$\begin{aligned} Z_{local} := &\{(s_{i,j}, a_i) \mid \exists\, a \in A_{logic}, s_i \in S_{logic} \colon (s_i, a) \in Z_{logic}\} \cup \\ &\{(a_i, s_{i,j}) \mid \exists\, a \in A_{logic}, s_i \in S_{logic} \colon (a, s_i) \in Z_{logic}\} \cup \\ &\{(\tilde{s}, a_i) \mid \exists\, a \in A_{logic}, \tilde{s} \in S_{logic} \backslash S_{extern} \colon (\tilde{s}, a) \in Z_{logic} \wedge a_i \in A_{local,a}\} \cup \\ &\{(a_i, \tilde{s}) \mid \exists\, a \in A_{logic}, \tilde{s} \in S_{logic} \backslash S_{extern} \colon (a, \tilde{s}) \in Z_{logic} \wedge a_i \in A_{local,a}\} \end{aligned}$$

Schließlich gilt

$$AB_{local} = AB_{logic}$$

Die Kardinalität der Mengen T_i ist abhängig von den existierenden Datenflüssen zwischen kommunizierenden Aktivitäten. Sie kann durchaus kleiner sein als die Menge aller theoretisch möglichen Datenflüsse zwischen verschiedenen Aktivitäten, die über modellierte Zugriffe miteinander verbunden sind und jeweils an mehreren Orten ausgeführt werden. Die Existenz dieser Datenflüsse ist abhängig vom Kontext einer modellierten Prozesslandschaft. Ihre Anzahl wird daher vom Modellierer festgelegt.
Für die Schnittstellen in der lokalen Sicht einer Prozesslandschaft muss sichergestellt werden, dass diese nicht aus der Menge der Schnittstellen $s_i \in S_{logic}$ ausgewählt werden, sondern bis auf die Elemente der Menge $S_{logic} \backslash S_{extern}$ neu definiert werden. Diese Elemente der Menge $S_{logic} \backslash S_{extern}$ repräsentieren die Randstellen der Prozesslandschaft (bei denen also entweder ihr Vor- oder ihr Nachbereich leer ist). Des Weiteren müssen die Teilmengen S_i, S_j, die jeweils verschiedene Schnittstellen aus der logischen Sicht repräsentieren, paarweise disjunkt sein. Die Anzahl der Schnittstellen pro

Teilmenge ist beschränkt durch die Anzahl der über sie kommunizierenden Aktivitätenpaare. Somit kann S_{local} formuliert werden als die Menge $S_{logic} \backslash S_{extern}$ zusammen mit der Vereinigung aller Schnittstellen zwischen je zwei miteinander kommunizierenden Aktivitätenpaare.
Alle Zugriffe $z \in Z_{local}$ entstehen durch (mehrfaches) Ersetzen der Zugriffe $z \in Z_{logic}$. Die Zugriffe auf Schnittstellen $\tilde{s} \in S_{logic} \backslash S_{extern}$, deren Vor- oder Nachbereich leer ist (also $\bullet\tilde{s} = \emptyset$ oder $\tilde{s}\bullet = \emptyset$), werden dabei gesondert behandelt, da sie nicht den Restriktionen der Definition 3.2.46 genügen. Hier werden die Zugriffe $(\tilde{s}, a)$ und $(a, \tilde{s})$ so oft ersetzt, wie es Ortsattribute der jeweils auf sie zugreifenden Aktivitäten $a \in A_{logic}$ gibt.
Auf den Aktivitätenbaum der Prozesslandschaft hat die Umstrukturierung in die lokale Sicht keinerlei Auswirkungen.

Die Anzahl der Schnittstellen in der lokalen Sicht einer Prozesslandschaft, die durch die Umstrukturierung entstehen, ist nach oben beschränkt. Formal:

Bemerkung 3.2.47. *Sei $PL_{logic} = (A_{logic}, S_{logic}, Z_{logic}, AB_{logic})$ die dritte Ausbaustufe einer nach logischen Gesichtspunkten modellierten Prozesslandschaft und $O \neq \emptyset$ eine gültige Ortemenge, deren Elemente mit Hilfe der Abbildung $local$ den Aktivitäten der Prozesslandschaft zugeordnet sind.*

Sei weiterhin $PL_{local} = (A_{local}, S_{local}, Z_{local}, AB_{local})$ die dritte Ausbaustufe einer nach lokalen Gesichtspunkten umstrukturierten Prozesslandschaft, die aus der Umstrukturierung von PL_{logic} entstanden ist und $a_1, a_2 \in A_{logic}, a_1 \neq a_2$. Es gilt:

$$\begin{aligned}
&\forall\, s \in S_{logic} \text{ mit } \bullet s = \{a_1\}, s\bullet = \{a_2\} : \\
&|\{s_k \in S_{local} \mid \exists\, o_i \in local(a_1) : (a_1, o_i) = \bullet s_k \vee \\
&\qquad\qquad \exists\, o_j \in local(a_2) : (a_2, o_j) = s_k\bullet\}| \leq |local(a_1)| \times |local(a_2)|
\end{aligned}$$

Mit den formulierten Restriktionen wird gefordert, dass in Abhängigkeit der Kardinalität der Aktivitätsteilmengen A_{local,a_1} und A_{local,a_2} auch deren Schnittstellen entsprechend häufig in der Prozesslandschaft PL_{local} vorkommen können: Die Anzahl der Schnittstellen $s \in S_{local}$ ist nach oben beschränkt durch das Produkt der Anzahl derjenigen Orte, an denen die Aktivität a_1 durchgeführt wird und der Anzahl derjenigen Orte, an denen die Aktivität a_2 durchgeführt wird. Die Menge der so entstehenden zusätzlichen Schnittstellen kann aber auch kleiner sein, wenn beispielsweise in einer zu modellierenden Prozesslandschaft nicht alle aus syntaktischer Sicht möglichen Schnittstellen tatsächlich existieren.

Externe Schnittstellen verbinden in der lokalen Sicht nicht mehr zwei verfeinerte Aktivitäten (vgl. Bemerkung 3.2.7 auf Seite 96), sondern sind genau diejenigen, über die zwei Aktivitäten unterschiedlicher Standorte Informationen austauschen. Damit gilt:

Konsistenzbedingung 3.2.48. *Sei $PL_{local} = (A_{local}, S_{local}, Z_{local}, AB_{local})$ die dritte Ausbaustufe einer nach lokalen Gesichtspunkten strukturierte Prozesslandschaft und $S_{local,extern} \subseteq S_{local}$ die Menge der externen Schnittstellen in PL_{local}.*

Sei weiterhin $\widetilde{local}$ eine Abbildung, die die Aktivitäten der lokalen Sicht auf ihr jeweiliges Ortsattribut $o \in O$ abbildet:

$$\widetilde{local} : A_{local} \rightarrow O \qquad \textit{mit } \widetilde{local}\big((a,o)\big) = o$$

$$\forall\, s \in S_{local} : \bigcup_{a \in \bullet s} \widetilde{local}(a) \neq \bigcup_{a \in s \bullet} \widetilde{local}(a) \Rightarrow s \in S_{local,extern}$$

Mit dieser Konsistenzbedingung wird die Menge der Schnittstellen, die die Restriktionen für externe Schnittstellen gemäß Definition 3.2.4 erfüllen, in der lokalen Sicht eingeschränkt auf diejenigen, die Aktivitäten zwischen verschiedenen Standorten miteinander verbinden. Diese Einschränkung hilft dem Analysten, der den Schwerpunkt seiner Analyse auf das Kommunikationsverhalten zwischen verschiedenen Standorten innerhalb einer Prozesslandschaft legen will.

Was diese Konsistenzbedingung sowie die Definition 3.2.46 für einen Modellierer bedeuten, der eine Umstrukturierung durchführen will, zeigt das nachfolgende Beispiel. Abbildung 3.29 zeigt einen Ausschnitt aus der Beispielprozesslandschaft PL_{logic} für komponentenbasierte Softwareentwicklung. Er beinhaltet Aktivitäten des Component Engineering, die nach der Komponentenentwicklung und der Releaseerstellung jede neue Komponente vom Qualitätsmanagement testen lassen. Die Aktivität *Fehlerkorrektur* erhält anschließend ein Fehlerprotokoll, dessen Inhalt darüber entscheidet, ob Mängel behoben werden müssen und die neue Komponente erneut dem Qualitätsmanagement vorzulegen ist oder aber diese an die Aktivität *Komponente hinzufügen* weitergeleitet werden kann. Die Aktivität *Fehlerkorrektur* hat damit ein deterministisches Ausgangsschaltverhalten. Dem Qualitätsmanagement ist ein deterministisches Eingangsschaltverhalten zugeordnet, da hier eine neue Komponente entweder erstmalig oder nochmalig getestet werden muss.

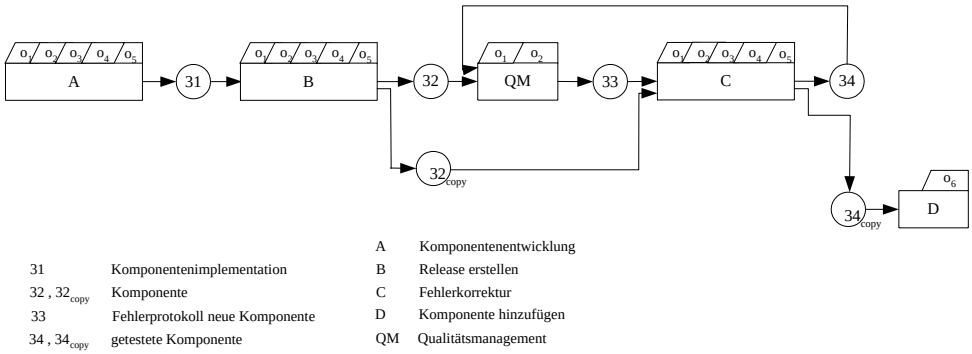


Abbildung 3.29: Ausschnitt der Beispielprozesslandschaft PL_{logic}

Für das Beispiel gilt weiterhin, dass alle Aktivitäten des Component Engineering an fünf verschiedenen Standorten o_1 bis o_5 stattfinden (vgl. Reiter an den Aktivitäten in

Abbildung 3.29), während das Qualitätsmanagement lediglich an den beiden Standorten o_1 und o_2 vorkommt. Mit Hilfe einer erweiterten Dokumentensicht (vgl. Abschnitt 3.1.2), in der für jede Schnittstelle $s \in S_{logic}$ die Zugriffe aller Aktivitäten eines jeden Standortes explizit aufgeführt sind, kann ein Modellierer zunächst – in Abhängigkeit des modellierten Kontextes – die tatsächlich vorkommenden Datenflüsse aus der Menge aller theoretisch möglichen auswählen. Abbildung 3.30 zeigt diese erweiterte Dokumentensicht für alle Schnittstellen aus Abbildung 3.29.

Abbildung 3.31 zeigt alle tatsächlich vorkommenden Datenflüsse innerhalb der Beispiellandschaft aus Abbildung 3.29 als Prozessfragmente. Hier wird deutlich, warum in der Konsistenzbedingung 3.2.46 auf Seite 129 die Anzahl der in PL_{local} vorkommenden Schnittstellen und Zugriffe auch kleiner als die Anzahl aller möglichen sein kann: Alle Aktivitäten des Component Engineering, die am Standort o_1 durchgeführt werden, kommunizieren ausschließlich mit dem Qualitätsmanagement am selben Standort. Analoges gilt für die Aktivitäten des Component Engineering am Standort o_2. Diese und die Aktivitäten des Component Engineering an den Standorten o_3, o_4 und o_5 kommunizieren ausschließlich mit dem Qualitätsmanagement am Standort o_2.

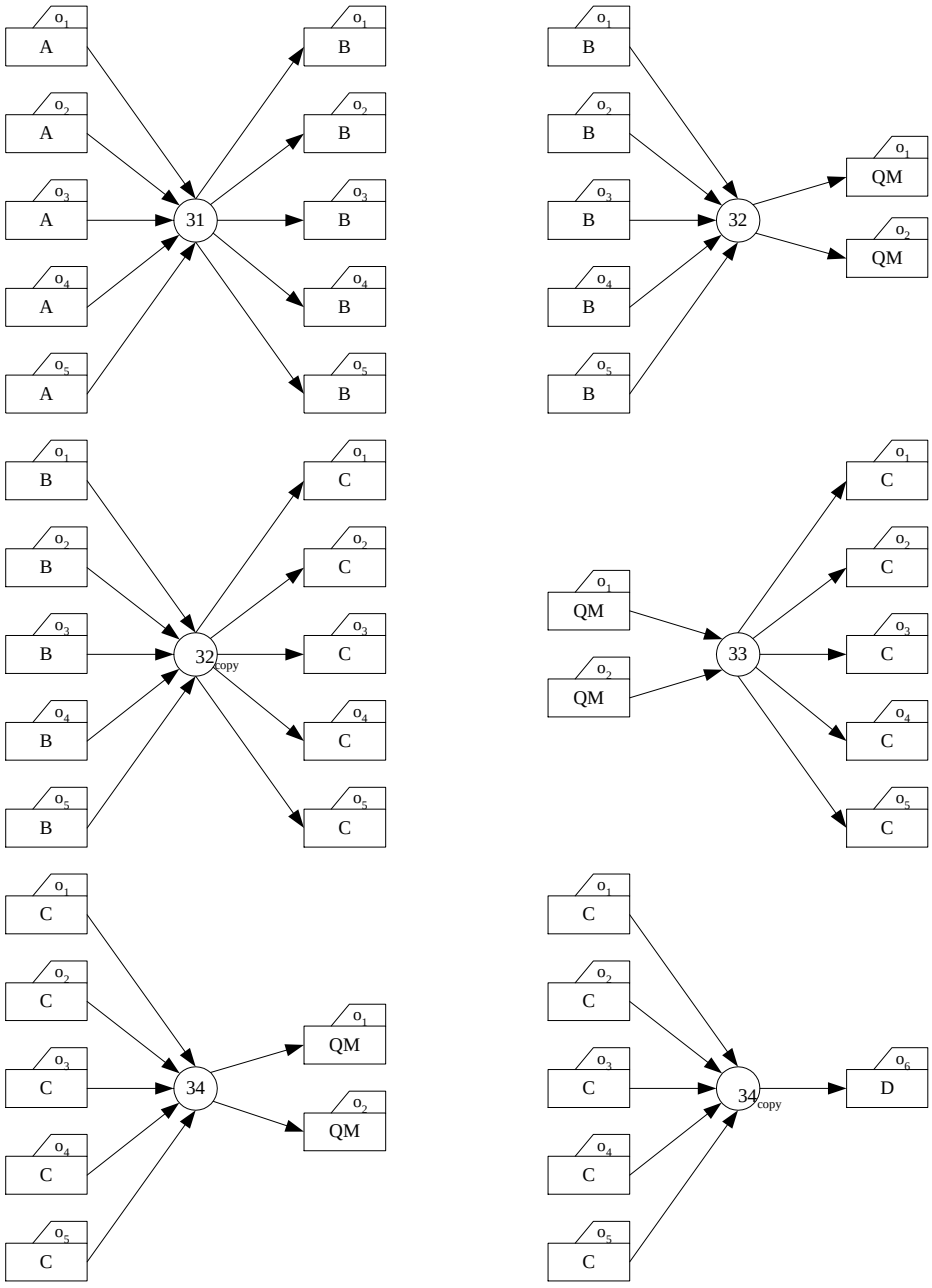
Verklebt man nun die Prozessfragmente aus Abbildung 3.31 derart, dass alle Aktivitäten mit gleichem Namen und gleichem Lokationsattribut jeweils nur einmal vorkommen, entsteht die Prozesslandschaft PL_{local} in Abbildung 3.32. Hier wird deutlich, was in Abbildung 3.29 nur durch zusätzliche Dokumentation der Lokationsattribute ersichtlich wurde. Die externen Schnittstellen sind außerhalb der grau hinterlegten Standorte o_1 bis o_6 angeordnet. Sie verbinden die insgesamt sechs vorhandenen Standorte der Prozesslandschaft miteinander und sind daher Grundlage für eine externe Kommunikation, die aus dieser lokalen Sicht fokussiert betrachtet werden kann.

Die Umstrukturierung einer Prozesslandschaft in eine lokale Sicht hat teilweise Auswirkungen auf die Schaltverhalten von Aktivitäten. Beispielsweise kann aus einem ursprünglich einfachen Ausgangsschaltverhalten jetzt ein deterministisches oder sogar komplexes Schaltverhalten geworden sein.

Insgesamt können mögliche Änderungen von Schaltverhalten nicht kontextunabhängig identifiziert und automatisiert durchgeführt werden. Dies ist Aufgabe des Modellierers, der in Abhängigkeit der modellierten Situationen das jeweilige Schaltverhalten anpassen muss. In Abbildung 3.32 ist unter anderem das Eingangsschaltverhalten der Aktivität *Komponente hinzufügen* von einem einfachen zu einem deterministischen verändert worden.

Mit der Anwendung der Abbildung $local$ auf alle Aktivitäten einer Prozesslandschaft PL_{logic}, ihrer Umstrukturierung und der anschließenden kontextabhängigen Anpassung verschiedener Schaltverhalten ist umfassend beschrieben, wie man die lokale Sicht für eine in einer logischen Sicht erstellten Prozesslandschaft erhält. Diese lokale Sicht kann analog der Verfeinerungen in der logischen Sicht weiter detailliert werden. Dazu werden alle beschriebenen Schritte individuell für jeden Ort der Prozesslandschaft erneut durchgeführt.

Neben der Möglichkeit einer weiteren Detaillierung wie gerade beschrieben sollte sich ein Modellierer zuvor jedoch immer die Frage nach möglichen Schnittstellen zwischen



31	Komponentenimplementation
32 , 32_{copy}	Komponente
33	Fehlerprotokoll neue Komponente
34 , 34_{copy}	getestete Komponente

A	Komponentenentwicklung
B	Release erstellen
C	Fehlerkorrektur
D	Komponente hinzufügen
QM	Qualitätsmanagement

Abbildung 3.30: Erweiterte Dokumentensicht der Schnittstellen aus Abbildung 3.29

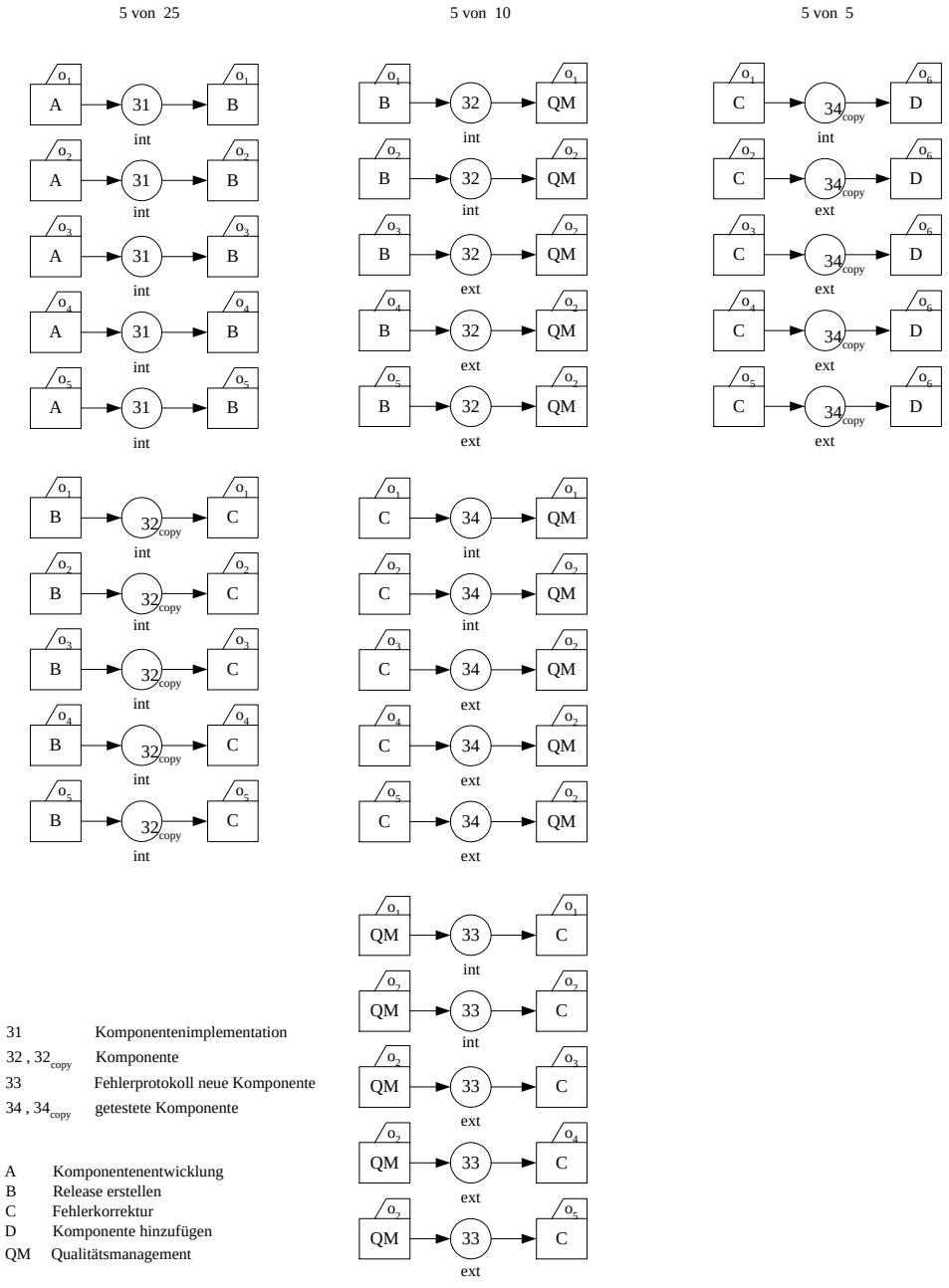


Abbildung 3.31: Auswahl der tatsächlich existierenden Datenflüsse

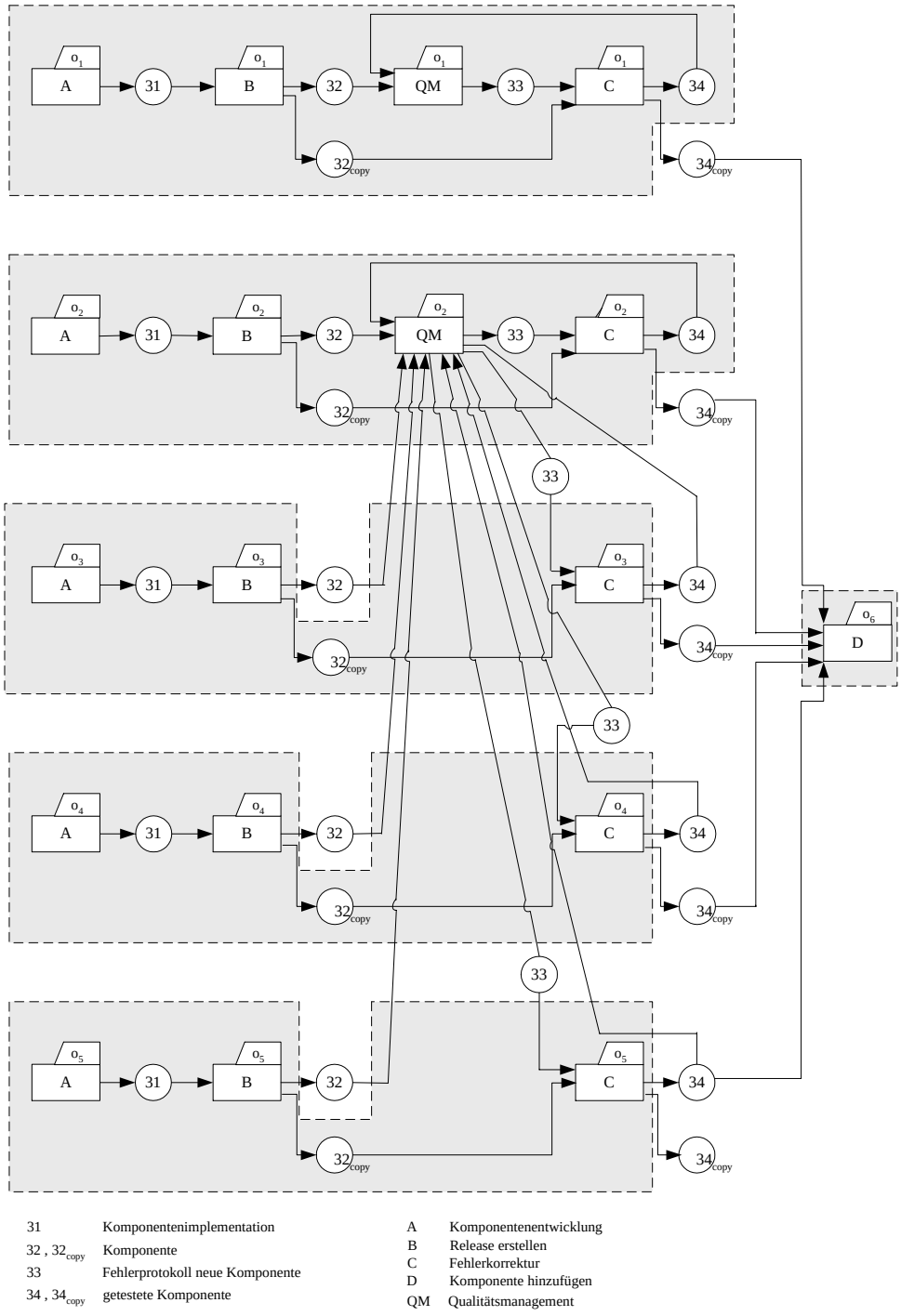


Abbildung 3.32: Lokale Sicht auf die Beispielprozesslandschaft aus Abbildung 3.29

denjenigen Aktivitäten stellen, die in der logischen Sicht nur einmal vorkommen. Beispielsweise sollten sich die Qualitätsmanagement-Aktivitäten der beiden Standorte o_1 und o_2 aus Abbildung 3.32 bezüglich der Vorgabe von Richtlinien abstimmen. Da auch die Frage nach weiteren Schnittstellen immer nur kontextabhängig beantwortet werden kann, bleibt diese Form der Erweiterung einer Prozesslandschaft ebenfalls Aufgabe des Modellierers und kann nicht automatisiert durchgeführt werden.

Für eine Prozesslandschaft PL_{bottom} werden im folgenden Abschnitt zusätzliche Erweiterungen und ausgezeichnete Teilnetze des korrespondierenden Petrinetzes eingeführt, die im Rahmen der Kommunikationsanalyse von unteren Landschaftsebenen erforderlich sind. Sie sind speziell bei der Berücksichtigung des geografischen Verteilungsaspektes – also der lokalen Sicht einer Prozesslandschaft – hilfreich, jedoch auch für die logische Sicht gültig.

3.2.2.2 Erweiterungen zur vierten Ausbaustufe einer Prozesslandschaft

Ziel der Analyse der unteren Ebenen einer verteilten Prozesslandschaft ist – wie bereits in Abschnitt 2.3.3 erläutert – die Effizienzbetrachtung des Kommunikationsverhaltens innerhalb dieser Landschaft. Das zugrundeliegende Petrinetz muss dazu die Festlegung und Auswertung verschiedener Landschaftsattribute ermöglichen, wie beispielsweise die Identifikation von Sender und Empfänger eines Informationsobjektes, dessen Volumen und entstehende Kommunikationskosten. Der Zusammenhang zwischen verschiedenen Aspekten der Effizienz, Landschaftseigenschaften und konkreten Attributen von Landschaftselementen ist in Abbildung 2.7 auf Seite 41 (Abschnitt 2.3.3) bereits informal dargestellt. Für die dort aufgeführten Attribute wird im Folgenden eine formale Basis eingeführt.

In PLL als Petrinetz-Variante ist jeder Schnittstelle ein nicht näher spezifizierter Dokumenttyp $dt \in DT$ zugewiesen (wie z.B. Anforderung, Review, etc., vgl. Definition 3.1.24 auf Seite 71). Für eine simulative Auswertung dynamischer Kommunikationseigenschaften über die verschiedenen repräsentierten Dokumente müssen diese Dokumenttypen mit Attributen versehen werden.

Beispiel:

Eine konkretes Dokument, das als Marke innerhalb einer Prozesslandschaft dargestellt wird, weist mindestens die folgenden Attribute auf:

- Sender
- Empfänger
- via (Kommunikationskanal, über den das Dokument verschickt wird)
- Id (zur eindeutigen Identifizierung)
- Volumen

Die Attributwerte für *Sender* und *Empfänger* entsprechen konkreten in der Prozesslandschaft modellierten Aktivitäten, die Attributwerte für *via* entsprechen konkreten Kommunikationskanälen. Eine Identitätsnummer und das Volumen eines Dokuments lassen sich jeweils als natürliche oder als reelle Zahl ausdrücken.

Formal kann ein Dokumenttyp $dt^* \in DT^*$ mit einer festen Menge an Attributen als kartesisches Produkt formuliert werden.

Definition 3.2.49. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft, $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle und DT^* eine Menge von Dokumenttypen mit $DT^* := A \times A \times CC \times \mathbb{Z} \times \mathbb{R}$. Seien weiterhin $a_1, a_2 \in A$, $cc_1 \in CC$, $n \in \mathbb{N}$ und $r \in \mathbb{R}$. Die Abbildung $\mathbf{type}^*$ ordnet den Schnittstellen von PL_{bottom} Dokumenttypen zu, die mit einer konkreten Attributmenge versehen sind:*

$$type^* : S \rightarrow DT^*$$

wobei für ein $dt^* \in DT^* gilt$:

$$dt^* := (a_1, a_2, cc_1, n, r)$$

Des Weiteren gilt:

- $netsender$ *ist eine Abbildung, die, angewendet auf ein $dt^* \in DT^*$, den Sender eines Dokumenttyps ermittelt:*

$$netsender : DT^* \rightarrow A_{local} \qquad \text{mit } netsender(dt^*) = a_1$$

- $netreceiver$ *ist eine Abbildung, die, angewendet auf ein $dt^* \in DT^*$, den Empfänger eines Dokumenttyps ermittelt:*

$$netreceiver : DT^* \rightarrow A_{local} \qquad \text{mit } netreceiver(dt^*) = a_2$$

- $netvia$ *ist eine Abbildung, die, angewendet auf ein $dt^* \in DT^*$, den zur Versendung eines Dokumenttyps verwendeten Kommunikationskanal ermittelt:*

$$netvia : DT^* \rightarrow CC \qquad \text{mit } netvia(dt^*) = cc_1$$

- $documentid$ *ist eine injektive Abbildung, die, angewendet auf ein $dt^* \in DT^*$, die Identitätsnummer eines Dokumenttyps ermittelt:*

$$documentid : DT^* \rightarrow \mathbb{N} \qquad \text{mit } documentid(dt^*) = n$$

- $volume$ *ist eine Abbildung, die, angewendet auf ein $dt^* \in DT^*$, das Volumen eines Dokumenttyps ermittelt:*

$$volume : DT^* \rightarrow \mathbb{R} \qquad \text{mit } volume(dt^*) = r$$

Die in Definition 3.2.49 eingeführte Konstruktion der Dokumenttypen erlaubt auf einfache Art und Weise die Auswertung von Simulationsdaten einer verteilten Prozesslandschaft bezüglich der Kommunikation verschiedenartig ausgeprägter Dokumenttypen.
Damit ist diejenige Teilmenge der in Abschnitt 2.3.3 geforderten Attribute für die Analyse spezieller Kommunikationseigenschaften definiert, die den zu betrachtenden Informationsobjekten zugeordnet werden kann. Nachfolgend werden zusätzliche Erweiterungen eingeführt, die der Betrachtung

- der Informationsverteilung,
- der räumlichen Verteilung von Aktivitäten,
- des Informationsflusses und
- der Auslastung aller modellierten Prozesse

dienen (vgl. Abschnitt 2.3.3) und damit eine Bewertung der Kommunikationseffizienz innerhalb einer Prozesslandschaft unterstützen.
Bei der Simulation wird ein Informationsaustausch dadurch modelliert, dass ein Kommunikationspartner, der Initiator, den Wunsch zur Kommunikation signalisiert. Erst wenn der andere Simulationspartner diese Anfrage beantwortet, übernimmt er dadurch die Rolle des Responders. Danach können Nachrichten zwischen beiden Partnern ausgetauscht werden. Dabei hängt die Richtung des Nachrichtenaustausches nicht zwingend mit der Rollenverteilung von Initiator und Responder zusammen.
Die Initiatorrolle ist formal dadurch gekennzeichnet, dass zum Vorbereich einer Aktivität eine ausgezeichnete Schnittstelle s_{init} existiert, die zur Schaltmenge dieser Aktivität gehört. Die Responderrolle lässt sich analog als ausgezeichnete Schnittstelle s_{resp} beschreiben, die zum Vorbereich der gleichen Aktivität gehört. Formal lassen sich die beiden Rollen wie folgt definieren:

Definition 3.2.50. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft mit $S = S_{intern} \cup S_{extern}$. Sei $S_{add} := S_{init} \cup S_{resp}$ mit $S_{init} \cap S_{resp} = \emptyset$ eine Menge von Schnittstellen, die der Prozesslandschaft PL_{bottom} zur Analyse des Informationsaustausches über Kommunikationskanäle hinzugefügt wurde.*
Sei weiterhin $CC \neq \emptyset$ die zugehörige Menge funktionsfähiger Kommunikationskanäle, a_1, $a_2 \in A$, $s_1, s_2 \in S_{extern}$ und $cc_1 := channel\big((a_1, s_1), (s_1, a_2)\big) \in CC$, $cc_2 := channel\big((a_1, s_2), (s_2, a_2)\big) \in CC$. Ein Element $s_{init} \in S_{init}$ heißt ***Initiatorrolle*** *für einen Informationsaustausch über einen Kommunikationskanal $cq :\Leftrightarrow$*

1. $\forall\, S_{source,a_1} \in (H_{source,a_1}) : s_{init} \in S_{source,a_1}$
2. $\bullet s_{init} = \emptyset$
3. $\forall\, a \in A\, \exists!\, s_{init} \in S_{init} : (s_{init}, a) \in Z$

*Ein Element $s_{resp} \in S_{resp}$ heißt **Responderrolle** für einen Informationsaustausch über einen Kommunikationskanal cc_2 :* $\Leftrightarrow$

1. $\forall\, S_{source,a_1} \in (H_{source,a_1}) : s_{resp} \in S_{source,a_1}$
2. $\bullet s_{resp} = \emptyset$
3. $\forall\, a \in A\, \exists!\, s_{resp} \in S_{resp} : (s_{resp}, a) \in Z$

Elemente aus S_{init} und S_{resp} sind damit immer Elemente der Schaltmengen einer Aktivität $a_1 \in A$, die wiederum einem Kommunikationskanal zugeordnet ist. Für eine passende Verteilung von Initiator- und Responderrolle muss gelten: $cc_1 = cc_2$ und damit $s_1 = s_2$.
Eine gültige Anfangsmarkierung als notwendige Voraussetzung zur Aktivierung des Kommunikationskanals cc lässt sich – bis auf die Schnittstelle, die die zu kommunizierende Nachricht enthält – analog zur gültigen Anfangsmarkierung innerhalb einer Prozesslandschaft PL formulieren (vgl. Definition 3.2.15 auf Seite 104). Die Bedingungen für s_{init} und s_{resp} müssen nicht notwendigerweise erfüllt sein, wenn keine Betrachtung der Informationsverteilung durchgeführt werden soll, die Initiator-/Responder-Rolle also nicht explizit betrachtet wird.

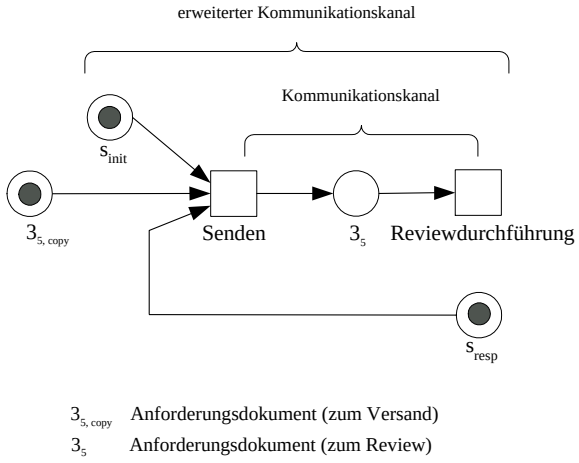


Abbildung 3.33: Kommunikationskanal erweitert um Initiator- und Responderrolle

Abbildung 3.33 zeigt beispielhaft den Aufbau des Kommunikationskanals, der ein Anforderungsdokument von der Aktivität *Senden* (vgl. Abbildung 3.21 auf Seite 116) über eine externe Schnittstelle an die Aktivität *Reviewdurchführung* des Qualitätsmanagements weiterleitet. Da nicht nur die Datenschnittstelle $3_{5,copy}$, sondern sowohl die Initiatorrolle s_{init} als auch die Responderstelle s_{resp} mit je einer Marke belegt ist, kann die Kommunikation über die Schnittstelle *Anforderungsdokument (zum Review)* unmittelbar stattfinden. Von der konkreten Typisierung der Datenschnittstellen 3_5 und $3_{5,copy}$ ist abstrahiert worden.

Mit Hilfe der Erweiterung eines Kommunikationskanals durch eine Initiator- und eine Responderrolle ist sichergestellt, dass eine Datenübertragung über eine Schnittstelle s_1 nur genau dann stattfinden kann, wenn eine Markierungsfunktion den Schnittstellen s_{init} und s_{resp} mindestens je eine Marke zuordnet. Wie es zu dieser Belegung kommt, ist eine Designentscheidung des Modellierers. Eine permanente Sende-/Empfangsbereitschaft kann beispielsweise dadurch implementiert werden, dass diese Schnittstellen initial belegt sind und zu den Zugriffen (s_{init}, a) und (s_{resp}, a) jeweils zurückführende Zugriffe (a, s_{init}) und (a, s_{resp}) modelliert werden.

Für die Analyse dynamischer Kommunikationseigenschaften ist die Betrachtung der Initiator- und Responderrolle nützlich, da mit ihrer Hilfe der potentielle Nutzungsgrad bereitgestellter Informationen und damit die **Kommunikationseffizienz bezüglich der Informationsverteilung** gemessen werden kann. Da die Entwicklung eines konkreten Beispiels aufgrund der durchzuführenden Parametrisierung und Simulation einer Prozesslandschaft an dieser Stelle den Rahmen sprengen würde, sei hierfür auf Abschnitt 4.2.2 verwiesen, in dem die Informationsverteilung am Beispiel der Prozesslandschaft zur komponentenbasierten Softwareentwicklung ausführlich diskutiert wird.

Für eine Menge von Kommunikationskanälen können Kapazität und Kostenfaktor für ihre Nutzung durch zwei einfache Zuweisungsfunktionen formal ausgedrückt werden.

Definition 3.2.51. *Sei $CC \neq \emptyset$ eine Menge von funktionsfähigen Kommunikationskanälen. Sei weiterhin $CAP \subseteq \mathbb{R}^+$ eine Menge möglicher **Kanalkapazitäten** (z.B. gemessen in kBit pro Sekunde). Die Abbildung $\mathbf{capacity}$ ordnet jedem Kommunikationskanal $cc \in CC$ eine konkrete Datenkapazität zu:*

$$capacity : CC \rightarrow CAP$$

Definition 3.2.52. *Sei $CC \neq \emptyset$ eine Menge von funktionsfähigen Kommunikationskanälen. Sei weiterhin $COST \subseteq \mathbb{R}^+$ eine Menge möglicher **Kanalkosten** (z.B. gemessen in Euro). Die Abbildung $\mathbf{cost}$ ordnet jedem Kommunikationskanal $cc \in CC$ konkrete Kosten zu, die bei seiner Nutzung anfallen:*

$$cost : CC \rightarrow COST$$

Beispiel:

Ein einfaches Beispiel für Kosten und Kapazitäten eines Kommunikationskanals ist ein Faxgerät mit einer Kapazität von 14.000 Bit pro Sekunde, welches bei seiner Nutzung 0,12 Euro pro Minute an Kosten verursacht. Für eine reale Kostenberechnung werden jedoch meist komplexere Modelle verwendet, die das zu übertragende Volumen, die Kapazität des Kanals, eine Grundgebühr und die Nutzungsdauer berücksichtigen.

Im Rahmen der Process Landscaping-Analyse dynamischer Kommunikationseigenschaften wird die diskutierte Form der Zuweisung von Kapazitäten und Kostenmaßen an Kommunikationskanäle genutzt, um das Kommunikationsaufkommen in einer verteilten Prozesslandschaft zu messen und damit die Bewertung der **Kommunikationseffizienz bezüglich der räumlichen Verteilung** der beteiligten Aktivitäten durchführen zu können. Sie gilt als gut, wenn die Zuordnung von modellierten Prozessen

zu einzelnen Orten zumindest für einen Großteil ($> 50\%$) der beteiligten Orte zu einem niedrigen Verhältnis von externem zu internem Kommunikationsaufkommen führt (vgl. Abschnitt 2.3.5). Für die Diskussion eines konkreten Beispiels sei auch hier wieder auf Abschnitt 4.2.2 verwiesen.

Um zeitabhängiges Ablaufverhalten innerhalb einer Prozesslandschaft analysieren zu können, kann das zugrundeliegende Petrinetz um Zeitstempel erweitert werden. Diese Zeitstempel beschreiben bezüglich einer globalen Uhr für die gesamte Prozesslandschaft den frühesten Zeitpunkt, zu dem ein Informationsobjekt zum Schalten einer Aktivität zur Verfügung steht. Die Werte der globalen Uhr repräsentieren damit Modellzeiten, die bei der Ausführung einer Prozesslandschaft bzw. eines sie repräsentierenden Petrinetzes berücksichtigt werden.

Definition 3.2.53. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft und $\mathbb{Z}$ die Menge der ganzen Zahlen. Die Abbildung* $\mathbf{timestamp}$ *definiert den Zeitraum, den eine Schnittstelle mit einer Marke belegt ist, bevor letztere als aktivierende Marke im Vorbereich einer Aktivität angesehen wird:*

$$timestamp : S \rightarrow \mathbb{Z}$$

Konkret bedeutet beispielsweise der Wert 10, dass zu einem Zeitpunkt $t = 100$ eine Aktivität zwar ein Informationsobjekt erzeugt und im Nachbereich ablegt, dieses aber erst zum Zeitpunkt $t = 110$ für nachfolgende Aktivitäten zur Verfügung steht.

Eine Zeitbehaftung lässt sich bei Petrinetzen aber auch anders recht einfach beschreiben: Jeder Ausgabekante $z = (a, s) \in Z$ kann ein Ausdruck zugeordnet werden, der einen Zahlenwert (integer) liefert. Dieser gibt, in Bezug auf eine globale Uhr, den frühesten Zeitpunkt an, zu dem ein Informationsobjekt zum Schalten einer Aktivität zur Verfügung steht. Prinzipiell ist jedes Netzelement geeignet, mit Zeitattributen versehen zu werden [Bau96].

Der Zeitfaktor spielt besonders bei der Analyse der **Kommunikationseffizienz bezüglich der Prozessauslastung** eine wichtige Rolle. Die Auslastung eines Prozesses ist in Abschnitt 2.3.5 definiert als das Verhältnis der Zeitspanne, in der mindestens eine Aktivität eines Prozesses aktiv ist, zur Gesamtzeit der Betrachtung.

Eine weitere Effizienzbetrachtung im Rahmen der Analyse dynamischer Kommunikationseigenschaften bezieht sich auf den **Informationsfluss innerhalb einer verteilten Prozesslandschaft**. Dieser sollte für jedes betrachtete Informationsobjekt eine Mindestlänge nicht unterschreiten, um sogenannte Ping-Pong-Kommunikation [GW99b] zu vermeiden.

Für die Berechnung der Länge eines Informationsflusses, nachfolgend als Pfadlänge bezeichnet, wird die Pfadlänge eines konkreten Informationstyps berechnet, den dieser, ausgehend von einer Aktivität a_1 an einem Ort o_1, über verschiedene Aktivitäten an einem Ort o_2 zurücklegt, bevor er zur Aktivität a_{n+1} an Ort o_1 zurückgelangt. Im korrespondierenden Petrinetz müssen hierfür zwei externe Schnittstellen $s_1, s_n \in S_{extern}$ mit dem gleichen Typ assoziiert sein. Eine Pfadlänge berechnet sich aus der Anzahl aller Aktivitäten und Schnittstellen, die der betrachtete Informationstyp auf seinem Weg von einer externen Schnittstelle zur anderen passiert.

Definition 3.2.54. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft. Ein **Pfad p** innerhalb einer Prozesslandschaft PL_{bottom} von einer Aktivität a_1 zu einer Aktivität a_{n+1} ist definiert als Folge von Aktivitäten und Schnittstellen:*

$$p := (a_1, s_1, a_2, s_2, ..., s_n, a_{n+1})$$

Dabei gilt: $a_i \in A$ *mit* $1 \leq i \leq n+1$,
$s_i \in S$ *mit* $1 \leq i \leq n$ *und*
$(a_1, s_1), (s_1, a_2), (a_2, s_2), ..., (s_n, a_{n+1}) \in Z$

Im Folgenden seien PL_{o_1} und PL_{o_2} zwei beliebige aber feste Ausschnitte einer Prozesslandschaft $PL_{bottom} = (A, S, Z, AB)$ mit $S = S_{intern} \cup S_{extern}$ und $o_1, o_2 \in O$, für die gilt, dass

- $PL_{o_1} = (A_{o_1}, S_{o_1}, Z_{o_1}, AB)$ mit $S_{o_1} = S_{intern_{o_1}} \cup S_{extern_{o_1}}$
- $PL_{o_2} = (A_{o_2}, S_{o_2}, Z_{o_2}, AB)$ und $S_{o_2} = S_{intern_{o_2}} \cup S_{extern_{o_2}}$ mit
  - $A_{o_1}, A_{o_2} \subset A$, $A_{o_1} \cap A_{o_2} = \emptyset$ und
    $\forall\, a \in A_{o_1} : local(a) = \{o_1\}$
    $\forall\, a \in A_{o_2} : local(a) = \{o_2\}$
  - $S_{intern_{o_1}}, S_{intern_{o_2}} \subset S_{intern}$ und $S_{o_1} \cap S_{o_2} = \emptyset$
  - $S_{extern_{o_1}}, S_{extern_{o_2}} \subset S_{extern}$
  - $Z_{o_1}, Z_{o_2} \subset Z$ und $Z_{o_1} \cap Z_{o_2} = \emptyset$

Definition 3.2.55. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft und $S = S_{intern} \cup S_{extern}$. Seien weiterhin*

- $type^* : S \rightarrow DT^*$ *die in Definition 3.2.49 eingeführte Abbildung, die den Schnittstellen einer Prozesslandschaft Dokumenttypen zuordnet und*
- $documentid : DT^* \rightarrow \mathbb{N}$ *die ebenfalls in Definition 3.2.49 eingeführte Abbildung, die die Identitätsnummer eines Dokumenttyps ermittelt.*

Schließlich seien $s_1, s_n \in S_{extern}$, für die gilt:

$$documentid\big(type^*(s_1)\big) = documentid\big(type^*(s_n)\big)$$

*Die **Länge eines Pfades** $p = (a_1, s_1, a_2, s_2, ..., s_n, a_{n+1})$ mit $n \in \mathbb{N}$ innerhalb einer Prozesslandschaft PL_{o_2}, der bei einer Aktivität $a_1 \in A_{o_1}$ startet und bei einer Aktivität $a_{n+1} \in A_{o_1}$ endet, aber ansonsten vollständig in PL_{o_2} liegt, ist definiert als*

$$pathlength(p) := 2n - 2$$

Für die Aktivitäten und Schnittstellen auf einem solchen Pfad p müssen dabei die folgenden Bedingungen erfüllt sein:

$$a_1, a_{n+1} \in A_{o_1}$$
$$s_1, s_n \in S_{extern}$$
$$a_2, a_3..., a_n \in A_{o_2}$$
$$s_2, ..., s_{n-1} \in S_{intern_{o_2}}$$
$$(a_1, s_1), (s_n, a_{n+1}) \in Z_{o_1}$$
$$(s_1, a_2), (a_2, s_2), ..., (s_{n-1}, a_n) \in Z_{o_2}$$

Bemerkung 3.2.56. *Die Länge eines Pfades p lässt die Zugriffe (a_1, s_1) und (s_n, a_{n+1}) außer acht, da laut Definition nur derjeinige Teil des Pfades betrachtet wird, der vollständig in PL_{o_2} liegt.*

Beispiel:

Abbildung 3.34 zeigt einen relativ kurzen Pfad des Informationstyps *Richtlinienerweiterung*, der als Vorschlag vom Qualitätsmanagement über eine verfeinerte Schnittstelle an das Projektmanagement weitergeleitet, dort geprüft und beurteilt wird und anschließend mit einem entsprechenden Vermerk an das Qualitätsmanagement zurückgeschickt wird.

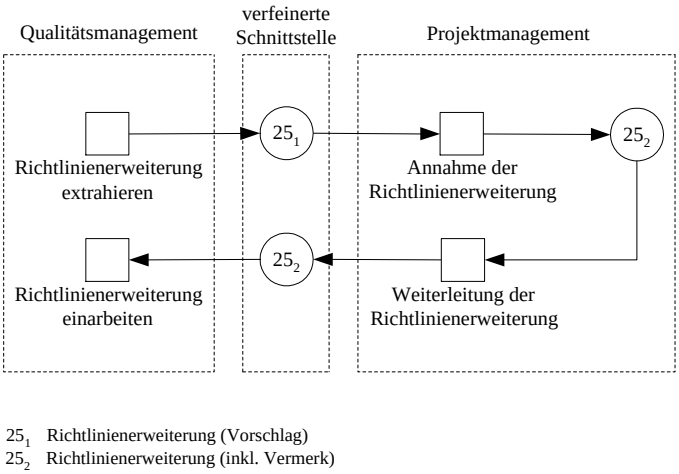


Abbildung 3.34: Pfad des Informationstyps *Richtlinienerweiterung*

Der betrachtete Pfad p kann formuliert werden als p = (*Richtlinienerweiterung extrahieren, Richtlinienerweiterung (Vorschlag), Annahme der Richtlinienerweiterung, Richtlinienerweiterung (inkl. Vermerk), Weiterleitung der Richtlinienerweiterung, Richtlinienerweiterung (inkl. Vermerk), Richtlinienerweiterung einarbeiten*). Er startet am Standort *Qualitätsmanagement*, der die Prozesslandschaft PL_{o_1} repräsentiert, durchläuft einen Teil des Standortes *Projektmanagement* (PL_{o_2}) und endet

schließlich wieder am Standort *Qualitätsmanagement*. Alle in Abbildung 3.34 dargestellten Schnittstellen sind mit der Nummer 25 bezeichnet. Dies deutet an, dass der dort dargestellte Pfad ausschließlich den des Informationstyps *Richtlinienerweiterung* repräsentiert. Damit gilt: $documentid(Richtlinienerweiterung(Vorschlag)) = documentid(Richtlinienerweiterung(inkl.Vermerk))$.

Die Pfadlänge $pathlength(p)$ beträgt in diesem Fall vier, da die Zugriffe (*Richtlinienerweiterung extrahieren, Richtlinienerweiterung (Vorschlag)*) und (*Richtlinienerweiterung (inkl. Vermerk), Richtlinienerweiterung einarbeiten*) bei der Zählung nicht berücksichtigt werden.

Notation 3.2.57. *Der längste mögliche Pfad innerhalb einer Prozesslandschaft PL_{o_2}, der bei einer Aktivität $a_1 \in A_{o_1}$ startet, über die Schnittstelle s_1 zur Prozesslandschaft PL_{o_2} verläuft, über die Schnittstelle s_n zur Prozesslandschaft PL_{o_1} zurückkehrt und bei einer Aktivität $a_{n+1} \in A_{o_1}$ endet, aber ansonsten vollständig in PL_{o_2} liegt, wird mit*

$$\mathbf{maxpath}(\mathbf{a_1}, \mathbf{s_1}, \mathbf{s_n})$$

bezeichnet.

Der kürzeste mögliche Pfad innerhalb einer Prozesslandschaft PL_{o_2}, der bei einer Aktivität $a_1 \in A_{o_1}$ startet, über die Schnittstelle s_1 zur Prozesslandschaft PL_{o_2} verläuft, über die Schnittstelle s_n zur Prozesslandschaft PL_{o_1} zurückkehrt und bei einer Aktivität $a_{n+1} \in A_{o_1}$ endet, aber ansonsten vollständig in PL_{o_2} liegt, wird mit

$$\mathbf{minpath}(\mathbf{a_1}, \mathbf{s_1}, \mathbf{s_n})$$

bezeichnet.

Zur Analyse der Kommunikationseffizienz bezüglich der Informationsflüsse in einer Prozesslandschaft können die Pfadlängen verschiedener Informationstypen am Minimum und Maximum aller möglichen Pfade gemessen und bewertet werden (vgl. Abschnitt 2.3.5). Allgemein gilt dabei, dass eine kurze Pfadlänge zwischen Aktivitäten verschiedener Orte auf Verbesserungspotential hinweist, soweit sie nicht aufgrund bestimmter Aufgabenverteilungen (z.B. Controlling-Aufgaben) explizit gewünscht ist. In Abschnitt 4.2.2 wird dies am konkreten Beispiel diskutiert.

Mit den Erweiterungen einer Prozesslandschaft zur Analyse verschiedener, die Effizienz beeinflussenden Eigenschaften kann die nächste Ausbaustufe einer Prozesslandschaft formuliert werden.

Definition 3.2.58. *Sei $PL_{bottom} = (A, S, Z, AB)$ die dritte Ausbaustufe einer Prozesslandschaft. Sei weiterhin*

- *DT^* eine Menge von Dokumenttypen mit einer festen Anzahl an Attributen,*
- *$type^*$ eine Abbildung zur Zuordnung von Dokumenttypen $dt^* \in DT^*$ zu Schnittstellen $s \in S$,*
- *S_{init} eine Menge von Initiatorrollen,*

- S_{resp} *eine Menge von Responderrollen,*
- $CAP \subseteq \mathbb{R}^+$ *eine Menge möglicher Kanalkapazitäten,*
- $COST \subseteq \mathbb{R}^+$ *eine Menge möglicher Kanalkosten,*
- P *eine Menge von Pfaden* p *(gemäß Definition 3.2.54) innerhalb der Prozesslandschaft* PL_{bottom}*,*
- $capacity$ *eine Abbildung zur Festlegung von Kanalkapazitäten gemäß Definition 3.2.51,*
- $cost$ *eine Abbildung zur Festlegung von Kanalkosten gemäß Definition 3.2.52,*
- $timestamp$ *eine Abbildung zur Festlegung von Aktivitätsdauern gemäß Definition 3.2.53 und schließlich*
- $pathlength$ *eine Abbildung zur Bestimmung der Länge eines Pfades bezüglich zweier fester Landschaftsausschnitte* PL_{o_1} *und* PL_{o_2} *für einen konkreten Informationstyp.*

*Die **vierte Ausbaustufe einer Prozesslandschaft** ist definiert als ein Tupel*

$$\begin{aligned} PL_{com} := (&A, S, Z, AB, NAME, O, name, local, switch, ZZ, CC, K, per, synch, \\ &change, coded, priv, mult, channel, value_Z, M, M', f_{source}, f_{dest}, \\ &DT^*, S_{init}, S_{resp}, CAP, COST, P, type^*, capacity, cost, timestamp, \\ &pathlength) \end{aligned}$$

Bei dieser Ausbaustufe ist zu beachten, dass – im Gegensatz zu allen vorangegangenen Ausbaustufen – hier neben einer Erweiterung der Prozesslandschaft auch eine Ersetzung vorgenommen worden ist: die Menge DT der Dokumenttypen und die Abbildung $type$ sind durch die Menge DT^* der Dokumenttypen und die Abbildung $type^*$ ersetzt worden, um detailliertere Analysen durchführen zu können.

Mit der vierten Ausbaustufe einer Prozesslandschaft ist die formale Grundlage aller in Abschnitt 2.3.3 diskutierten Analyseziele der Methode des Process Landscaping beschrieben. Am Beispiel einer Prozesslandschaft zur komponentenbasierten Softwareentwicklung kann diese nun eingesetzt werden, um die verschiedenen Analyseziele zu validieren.

3.3 Einordnung in den Bereich der Petrinetze

Die Petrinetze als formale Basis des Process Landscaping haben sich als geeignetes Instrument zur Beschreibung einer verteilten Prozesslandschaft erwiesen: Alle in der Einleitung dieses Kapitels 3 aufgeführten Modellierungsanforderungen konnten erfüllt werden; ihre Umsetzung ist jeweils an Ausschnitten eines durchgängigen Beispiels

gezeigt worden. Für die Erstellung einer Prozesslandschaft nach dem Top-Down-Vorgehen sind Eigenschaften eingeführt worden, die bereits aus verschiedenen existierenden Petrinetz-Varianten bekannt sind. So konnte jeweils eine Prozesslandschaft – ausgehend von ihrer Basisdefinition – in ihrer ersten bis zu ihrer vierten Ausbaustufe formuliert werden.

Innerhalb der letzten 40 Jahre ist eine Vielzahl von Petrinetz-Varianten sowie Petrinetz-basierten Methoden und Werkzeugen entstanden. Sie werden in verschiedenen Bereichen wie beispielsweise technischen Steuerungssystemen und Workflowmanagementsystemen eingesetzt. Viele dieser Petrinetz-Varianten sind spezielle Erweiterungen zur Betrachtung spezifischer Eigenschaften eines modellierten Systems, oft inklusive der Betrachtung des dynamischen Verhaltens dieses Systems. Beispiele hierfür sind unter anderem die Verhaltensspezifikation von CORBA-Systemen [BPS+99] oder die Berücksichtigung unscharfer Zeitaussagen bei der Spezifikation von Hypermedia-Anwendungen [Alf03].

Alle Petrinetz-Formalismen bieten den Vorteil, dass die formal spezifizierten Prozessmodelle durch verschiedene Analyse- und Konsistenzbedingungen validiert und teilweise auch verifiziert werden können. Nachteile der herkömmlichen Petrinetz-Formalismen sind jedoch die hohe Komplexität und Größe schon bei sehr kleinen Ausschnitten eines realen Prozessmodells [BBD+99]. Gerade die Komplexität konnte durch die im Rahmen dieser Arbeit entwickelte abstrakte Petrinetz-Variante PLL verringert werden, die speziell zur Darstellung der oberen Ebenen einer Prozesslandschaft entwickelt worden ist. Der Begriff *abstrakt* weist an dieser Stelle auf die Abstraktion von Ablaufinformationen hin. Er wird damit anders als beispielsweise bei Padberg genutzt [Pad98]. Dort wird unter dem Konzept abstrakter Petrinetze die Instanziierung verschiedener Netzklassen verstanden, bei denen Datentyp und Netzstruktur als abstrakte Parameter angesehen werden. Die angesprochene Komplexität in der Darstellung eines Petrinetzes konnte in PLL dadurch verringert werden, dass kein zusammenhängendes Netz für eine Prozesslandschaft gefordert wird, sondern lediglich pro Abstraktionsebene alle beteiligten Aktivitäten, Schnittstellen und Zugriffe darzustellen sind. Der Zusammenhang aller Aktivitäten einer Prozesslandschaft bleibt über seinen Aktivitätenbaum transparent. Somit ist ein Teil der Komplexität durch die Auslagerung von Informationen – über den Zusammenhang von und die hierarchischen Beziehungen zwischen Aktivitäten – in den Aktivitätenbaum verringert worden. Die Darstellung der Dynamik eines Netzes entfällt bei PLL völlig, was die Komplexität noch weiter verringert. Auf diese Weise wird zwar die Aussagekraft vermindert, bei traditionellen Petrinetzen mögliche Aussagen über den Ablauf eines Prozesses sind jedoch hier gar nicht gefragt.

Das Vorgehen bei der Entwicklung und Anwendung verschiedener und damit auch verschieden komplexer Petrinetz-Varianten zur Modellierung und Analyse einer Prozesslandschaft mit der Methode des Process Landscaping entspricht den Strukturierungsmerkmalen des *Petrinetz-Baukastens* [GE01], einer Arbeit der DFG-Forschergruppe *Petrinetz-Technologie*. Dieser bietet die Möglichkeit, abhängig von Anforderungen an ein Petrinetz und einem Anwendungsbereich eine geeignete Petrinetz-Vari-

ante mit – soweit vorhanden – entsprechender Werkzeugunterstützung auszuwählen. Dazu sind von der DFG-Forschergruppe drei verschiedene Sichten auf Petrinetze entwickelt worden: die Anwendungsentwickler-Sicht, die Experten-Sicht und die Werkzeugentwickler-Sicht.

- Die Anwendungsentwickler-Sicht hilft einem Anwendungsentwickler bei der Suche nach einer Petrinetz-Notation, mit der er seine Anwendungsdomäne geeignet darstellen kann und die ihn beim Entwicklungsprozess seiner Applikation unterstützt.
- Die Experten-Sicht ermöglicht es einem Petrinetz-Experten, neue Typen und Netzeigenschaften in einer einheitlichen formalen Struktur zu definieren, von verschiedenen Netztypen zu profitieren und seine Ergebnisse dem Anwendungsentwickler geeignet zur Verfügung zu stellen.
- Die Werkzeugentwickler-Sicht erlaubt es schließlich einem Entwickler von Software-Werkzeugen, passende Werkzeuge für bestimmte Aufgaben zu finden und sie in geeigneter Form zu integrieren.

Eine semi-formale Petrinetz-Klassifikation bildet die gemeinsame Basis dieser drei Sichten. Diese Klassifikation beinhaltet verschiedene Petrinetz-Typen, die über ihre Attribute und Attributwerte strukturiert werden und beschreibt in einer objektorientierten Form die Relationen zwischen den verschiedenen Netz-Typen. Abbildung 3.35 [BBD+99] zeigt diese Klassifikation als Klassendiagramm in UML-Notation. Es dient hier als Diskussionsgrundlage zur Einordnung der für das Process Landscaping verwendeten Petrinetze bzw. der Eigenschaften, die sie mit verschiedenen existierenden Petrinetz-Typen gemeinsam haben.

Im Zentrum der Abbildung 3.35 steht die abstrakte Klasse *Petri Net Type* mit ihren acht Attributen *place*, *transition*, *tp-edge*, *pt-edge*, *activation*, *firing*, *marking* und *token*. Die ersten drei Attribute beschreiben die Existenz von Stellen und Transitionen als Bestandteile eines Petrinetzes sowie Kanten zwischen Stellen und Transitionen (*pt-edge*) und umgekehrt (*tp-edge*). Des Weiteren ist die Klasse durch die Existenz von Marken (*token*) auf Stellen, dem Aktivieren und Schalten von Transitionen und der Markierung von Stellen bzw. Netzen charakterisiert.

Alle Klassifikationsmerkmale für Petrinetze sind in Abbildung 3.35 als separate Klassen über eine Vererbungsrelation mit der abstrakten Klasse *Petri Net Type* verbunden. Sie werden als Spezialisierungen der Klasse *Petri Net Type* eingeführt, die selbst wiederum sogenannte Spezialisierungspfade enthalten. Diese werden teilweise im Folgenden detaillierter dargestellt, sofern dies der Diskussion um die Einordnung der im Rahmen des Process Landscaping genutzten Netzeigenschaften dient.

Klasse *1.1* repräsentiert das Klassifikationsmerkmal der Markierungsgrenze von Stellen und damit der maximal möglichen Anzahl an Marken auf einer Stelle. Diese kann entweder individuell unterschiedlich oder aber einheitlich festgelegt werden. Für die verschiedenen Ebenen einer Prozesslandschaft ist die Markierungsgrenze zwar auf einen einheitlichen, aber nicht auf den konkreten Wert 1 festgelegt worden. Das für

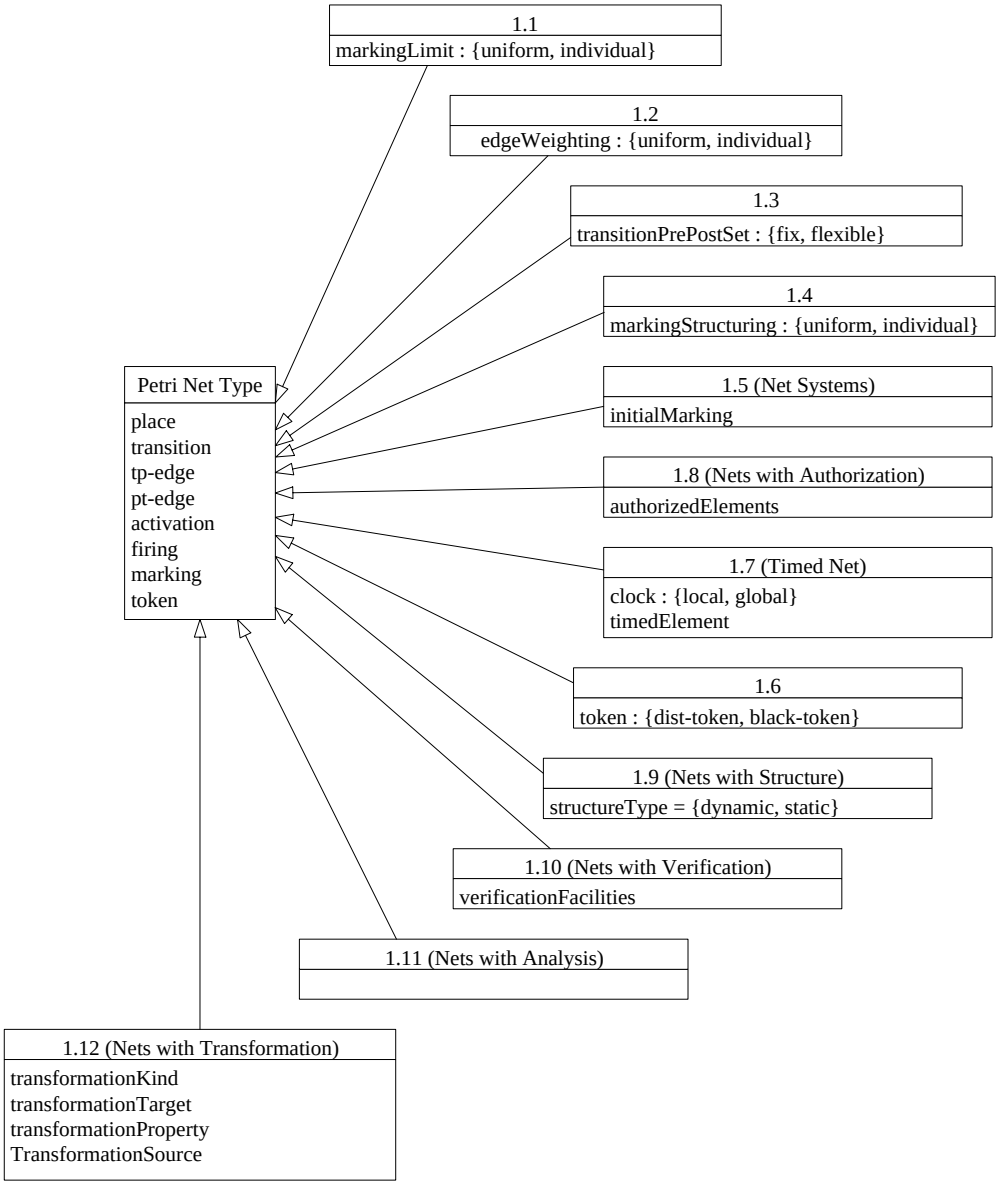


Abbildung 3.35: Objektorientierte Petrinetz-Klassifikation

eine Prozesslandschaft erlaubte Kantengewicht, in Abbildung 3.35 repräsentiert als Klasse *1.2*, ist für die Darstellung der unteren Ebenen auf bis unendlich groß festgelegt (vgl. Bemerkung 3.2.12, Seite 103).

Ein nächstes Klassifikationsmerkmal ist die Struktur des Vor- und Nachbereichs einer Transition (vgl. Klasse *1.3* in Abbildung 3.35). Diese ist für die oberen Ebenen einer Prozesslandschaft irrelevant und daher nicht näher spezifiziert; für die unteren Ebenen ist sie flexibel, da eine Transition sowohl unterschiedliche Eingangs- als auch Ausgangsschaltverhalten haben kann (vgl. Definitionen 3.2.27 und 3.2.28, Seite 111). Hier ist jedoch festzuhalten, dass die Definition der verschiedenen Schaltverhalten im Rahmen des Process Landscaping nicht von den Funsoft-Netzen [Gru91] übernommen worden sind. Die Gründe hierfür sind bereits in Abschnitt 3.2.1.2 auf Seite 103 erläutert worden. Zusätzlich ist zu den bislang bekannten Ausgangsschaltverhalten das abhängige Ausgangsschaltverhalten (vgl. Definition 3.2.26, Seite 111) neu eingeführt worden.

Klasse *1.4* repräsentiert das Klassifikationsmerkmal der Markierungsstruktur. Für die unteren Ebenen einer Prozesslandschaft ist diese einheitlich als ein Datentyp festgelegt, der als kartesisches Produkt von fünf Mengen formuliert ist (vgl. Definition 3.2.49, Seite 138). Mit der Existenz einer Markierungsstruktur gilt gleichzeitig, dass eine initiale Markierung für das den unteren Ebenen zugrundeliegende Petrinetz gegeben sein muss (vgl. Klasse *1.5*). Das Klassifikationsmerkmal der Autorisierung wird beim Process Landscaping dagegen nicht berücksichtigt (vgl. Klasse *1.8 (Nets with Authorization)*).

Zeitbehaftete Petrinetze (*Klasse 1.7 (Timed Net)*) haben mit dem einer Prozesslandschaft vierter Ausbaustufe zugrundeliegenden Petrinetz ebenfalls einige Eigenschaften gemeinsam, denn auf den unteren Ebenen einer Landschaft wird jeder Stelle ein Zeitstempel zugewiesen, der – bezogen auf eine globale Uhr – den frühesten Zeitpunkt angibt, zu dem ein Informationsobjekt zum Schalten einer Aktivität zur Verfügung steht (vgl. Definition 3.2.53, Seite 142).

Das Klassifikationsmerkmal der Marke, dargestellt mit der Klasse *1.6*, ist eng mit dem der Markierungsstruktur verknüpft, denn durch den Dokumenttyp, der mit einer festen Menge von Attributen versehen ist, werden für die unteren Ebenen einer Prozesslandschaft unterscheidbare Marken verwendet. Für ihre oberen Ebenen ist dieses Klassifikationsmerkmal nicht relevant. Abbildung 3.36 zeigt, dass – neben den bislang diskutierten – insbesondere dieses Merkmal direkt zur Verwendung von High-Level-Petrinetzen für die Darstellung der unteren Ebenen einer Prozesslandschaften führt, deren Datenstruktur explizit gegeben ist. Diese Netztypen zeichnen sich unter anderem dadurch aus, dass sie durch einfachere Petrinetze (Low-Level-Netze) ausgedrückt werden können [Smi98]. Durch die in der vierten Ausbaustufe einer Prozesslandschaft verwendete Stellentypisierung wird deutlich, dass das zugrundeliegende Petrinetz gemeinsame Merkmale mit Netzen höherer Ordnung und mit farbigen Petrinetzen hat (vgl. Klassen *1.6.1.1.1.2* und *1.6.1.1.1.3* in Abbildung 3.36). Das Process Landscaping verwendet hier zwar keine Inskriptionssprache wie beispielsweise die farbigen Netze [Jen97] und beschreibt die Datenstruktur des Dokumenttypen auch nicht als al-

gebraische Spezifikation wie die Prädikat-/Transitionsnetze [Gen86]. Trotzdem stützt sich die verwendete Beschreibung als kartesisches Produkt auf algebraische Grundlagen. Damit kann die vierte Ausbaustufe einer Prozesslandschaft als ähnlich zu bzw. verwandt mit den in Abbildung 3.36 durch die Klasse *1.6.1.1.1.2* repräsentierten algebraischen High-Level-Netzen angesehen werden.

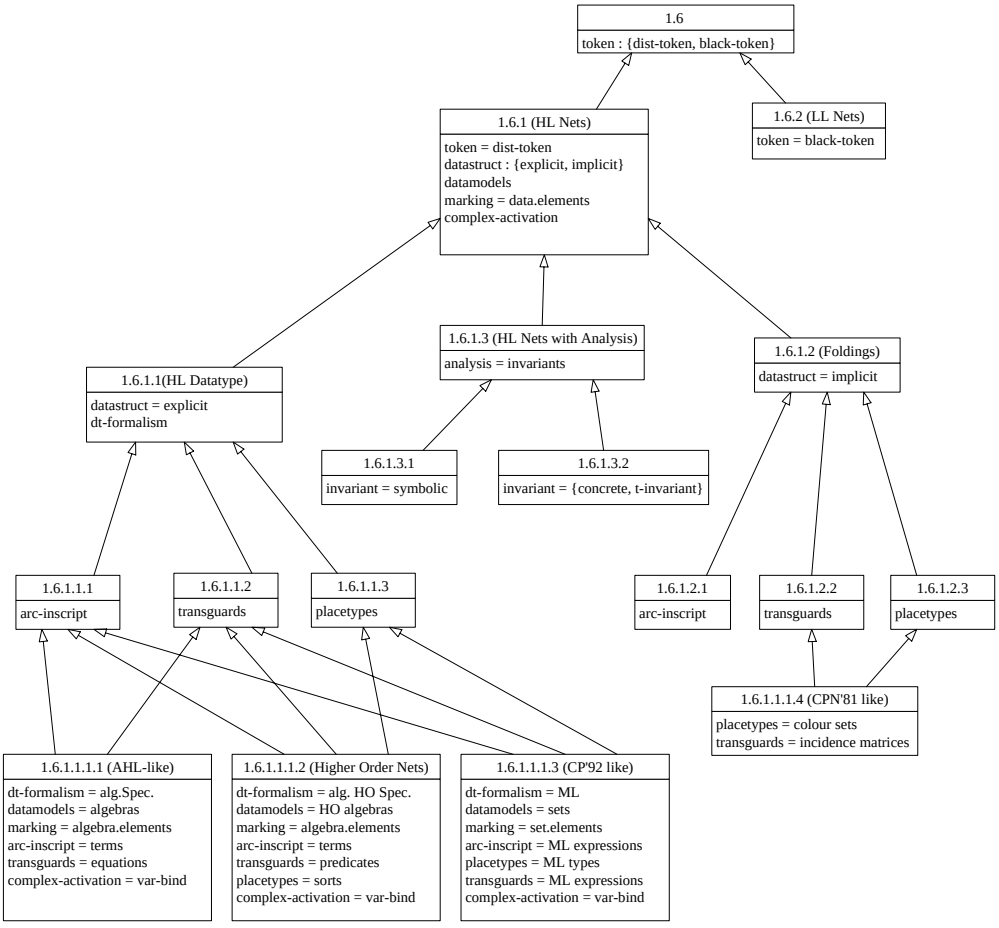


Abbildung 3.36: Klassifikationsmerkmal *Marke*

Bezüglich des Klassifikationsmerkmals der Netzstruktur (vgl. Klasse *1.9 (Nets with Structure)* in Abbildung 3.35) gilt für alle Ebenen einer Prozesslandschaft, dass das zugrundeliegende Petrinetz eine statische Struktur hat. Diese kann durch Verfeinerungen sowohl bezüglich einer Aktivität als auch einer Schnittstelle modifiziert werden. Das Konzept zur Verfeinerung einer Transition bzw. einer Stelle ist in der Literatur bereits bekannt [Bau96]. Für das Process Landscaping ist dieses jedoch verändert worden, aus Gründen, die in den Abschnitten 3.1.1 und 3.2.1.3 ausführlich diskutiert worden sind.

Die sukzessive Verfeinerung einer Prozesslandschaft führt gleichzeitig die Eigenschaft der Hierarchie für das zugrundeliegende Petrinetz ein. Diese Eigenschaft ist in der Li-

teratur bislang für High-Level-Netze wie beispielsweise den Funsoft-Netzen [Gru91] oder den hierarchischen farbigen Netzen [Jen97] bekannt. Im Rahmen des Process Landscaping wird sie jedoch bereits für die oberen Ebenen einer Prozesslandschaft eingeführt und damit eine Petrinetz-Variante – PLL –, die ansonsten keine Eigenschaft höherer Petrinetze aufweist. Während die bekannten High-Level-Netze Hierarchieinformationen über ausgezeichnete Transitionen und Stellen (substitution transitions, fusion places [Jen97]) darstellen, werden diese Informationen bei PLL durch den Aktivitätenbaum aufgezeigt. Letzterer wird für alle Ausbaustufen einer Prozesslandschaft genutzt, so dass die Darstellungsform der traditionellen hierarchischen Netze nicht mehr erforderlich ist.

Die beiden Klassifikationsmerkmale der Verifikation und der Analyse (vgl. Klassen *1.10 (Nets with Verification)* und *1.11 (Nets with Analysis)* in Abbildung 3.35) sind für das Process Landscaping nicht relevant. Statt der Verifikation und der Analyse von Netzstrukturen wie Lebendigkeit, Erreichbarkeit etc. werden konkrete Kommunikationseigenschaften einer Prozesslandschaft mittels numerischer Analyse oder Simulation validiert.

Das letzte der in Abbildung 3.35 aufgeführten Klassifikationsmerkmale ist das der Transformation (vgl. Klasse *1.12 (Nets with Transformation)*). Dieses Merkmal, in Abbildung 3.37 [BBD+99] detaillierter dargestellt, unterscheidet zwischen verschiedenen Formen einer Transformation. Sie starten meist von einem einfachen Petrinetz wie den Stellen-/Transitionsnetzen und werden über verschiedene Vorgehensweisen zu High-Level-Netzen erweitert (vgl. *transformationKind add* bzw. *recode* in Abbildung 3.37). Demgegenüber stehen andere Vorgehensweisen, die von einem High-Level-Netz ausgehen und dieses auf einfache Netze abbilden (vgl. *transformationKind ignore, encode* und *decode*). Das Process Landscaping folgt dem Prinzip, einem zunächst möglichst einfachen Petrinetztyp nach und nach Eigenschaften hinzuzufügen (*transformationKind add*), die schließlich zur vierten Ausbaustufe einer Prozesslandschaft mit einem zugrundeliegenden algebraischen High-Level-Netz führen.

Die in Abbildung 3.35 aufgeführten Klassifikationsmerkmale sind hauptsächlich für die unteren Ebenen einer Prozesslandschaft bzw. für die dritte und vierte Ausbaustufe einer Prozesslandschaft diskutiert worden. Dies liegt vor allem daran, dass mit PLL für die oberen Ebenen (erste und zweite Ausbaustufe) einer Prozesslandschaft aufgrund ihrer Abstraktion von Ablaufinformationen – und damit vom Kontrollfluss innerhalb eines zugrundeliegenden Petrinetzes – eine für Petrinetze sehr untypische Variante entstanden ist. In PLL ist gerade auf eine Stärke der Petrinetze – der Betrachtungsmöglichkeit dynamischer Systeme bzw. dynamischen Verhaltens auf Basis einer statischen Netzstruktur – explizit verzichtet worden. PLL ist damit nur schwer in das Klassifikationsschema einzuordnen. Dieser Netztyp kann auch nicht als abstrakte Variante eines einfachen Petrinetzes bezeichnet werden, da er mit seinem Verfeinerungskonzept für Aktivitäten und Schnittstellen sowie der daraus resultierenden hierarchischen Struktur typische Eigenschaften der High-Level-Netze aufweist.

Bezüglich der Darstellungsmöglichkeit für Informationsobjekte und Aktivitäten ist PLL mit Datenflussdiagrammen (DFD) [Bal96] vergleichbar. Bei Letzteren können

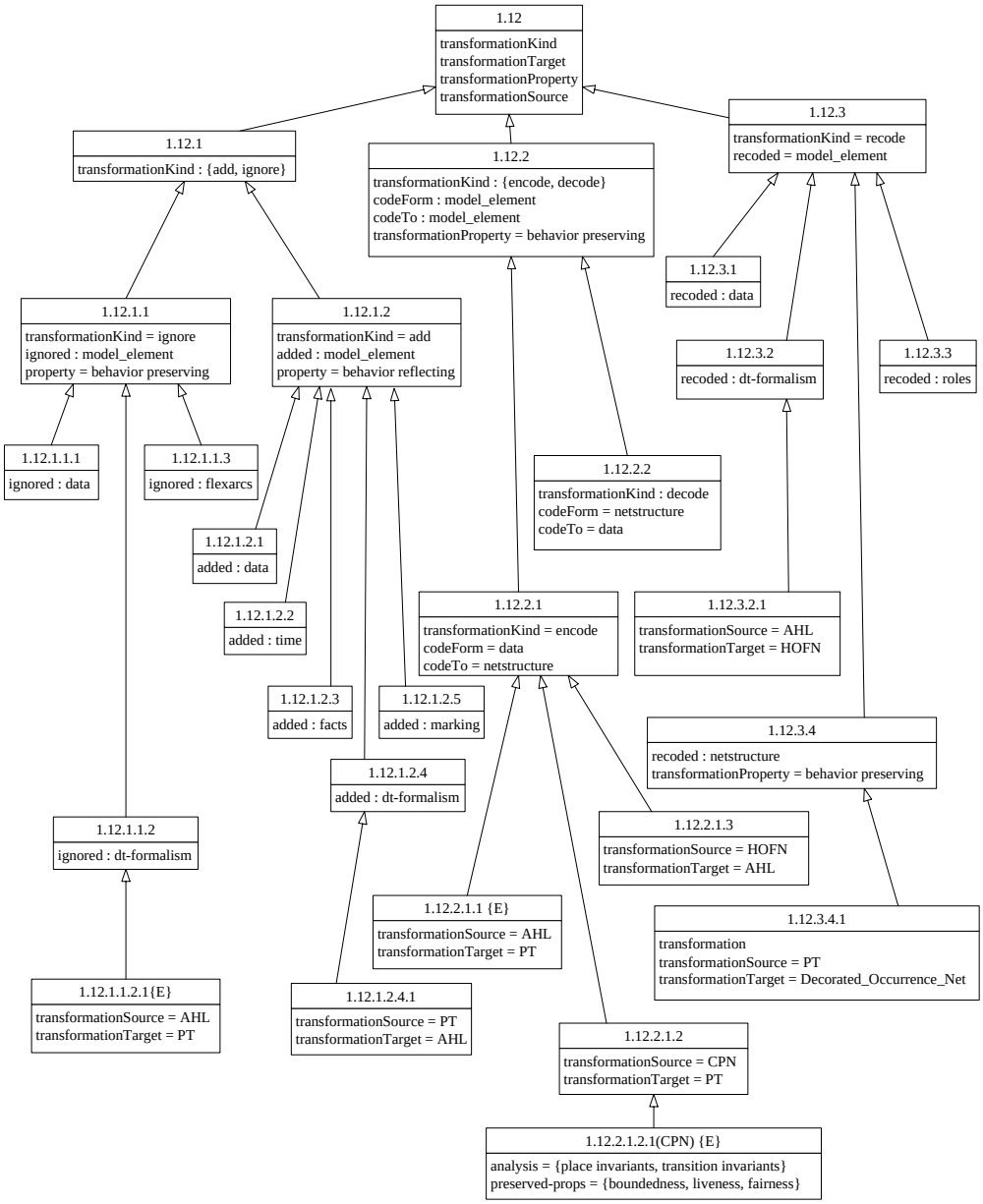


Abbildung 3.37: Klassifikationsmerkmal *Transformation*

die Funktionen als Aktivitäten einer Prozesslandschaft interpretiert werden. Zusätzlich gibt es die Möglichkeit der Modellierung von Datenspeichern, die jedoch für die obersten Ebenen einer Prozesslandschaft nicht von Bedeutung sind und daher keine Verwendung finden. Für die Wahl von PLL statt der DFD zur Modellierung der oberen Ebenen haben zwei Gründe gesprochen: Zum einen hätten die DFD mit größerem Aufwand um Verfeinerungskonzepte erweitert und angepasst werden müssen. Dies liegt unter anderem daran, dass bei einem DFD der Fokus eindeutig auf der Darstellung von Daten (in Form von Datenflüssen, Datenspeichern und Schnittstellen) liegt, während die Petrinetze den Fokus eher auf deren Verarbeitung legen (in Form von sequentiell und/oder parallel auszuführenden Aktivitäten). Ein zweiter Grund für die Entwicklung und den Einsatz von PLL ist die Verhinderung eines Notationsbruchs, der bei der Verwendung von DFDs für die oberen Ebenen einer Prozesslandschaft und dem Einsatz von Petrinetzen für die unteren Ebenen vorhanden wäre. Ein solcher Notationsbruch birgt immer die Gefahr von Inkonsistenzen. Mit PLL ist lediglich von – wenn auch typischen – Eigenschaften eines traditionellen Petrinetzes abstrahiert worden, die für den Übergang zur dritten Ausbaustufe einer Prozesslandschaft wieder hinzugefügt wurden.

Insgesamt hat sich somit bei der Suche nach einer geeigneten Petrinetz-Variante zur Unterstützung des Process Landscaping aus der Sicht eines Anwenders keine Variante aus dem Petrinetz-Baukasten gefunden, die den Anforderungen zur Erstellung der oberen Ebenen einer Prozesslandschaft genügt. Daher ist ein Wechsel zur Experten-Sicht erfolgt, um verschiedene Petrinetz-Typen zu entwickeln, die der Modellierung und Analyse von verteilten Prozesslandschaften unterschiedlicher Ausbaustufen dienen. Während die Methode, bedingt durch ihr Top-Down-Vorgehen, jedoch nacheinander alle eingeführten Petrinetz-Varianten nutzt, stellt der Petrinetz-Baukasten eine Sammlung verschiedener Varianten dar, aus denen sich ein Anwender die für ihn jeweils passende auswählen kann. Über die Klassifikation der Netztypen werden zwar ihre Relationen untereinander beschrieben, ein Übergang zwischen ihnen im Rahmen einer gemeinsamen Anwendung wird hier jedoch nicht unterstützt.

Eine Werkzeugunterstützung für die Anwendung der Methode des Process Landscaping ist bislang noch nicht diskutiert worden. Hierfür ist zunächst die Werkzeugentwickler-Sicht eingenommen worden, um aus der Menge der bereits vorhandenen Petrinetz-Werzeuge geeignete zu finden und diese zu integrieren. Die Werkzeugunterstützung ist jedoch Gegenstand des Kapitels 5 und wird daher an dieser Stelle nicht weiter diskutiert.

Kapitel 4

Validation der Analysemethoden

Neben der Modellierung einer verteilten Prozesslandschaft bildet die Analyse einen zweiten Schwerpunkt des Process Landscaping. Die Analysemethoden und ihre Ziele, die Untersuchung von Kommunikationseigenschaften einer Prozesslandschaft, sind bereits in Abschnitt 2.3 motiviert und erläutert worden. Die Modellierung einer Prozesslandschaft, die die Aktivitäten und Abläufe der komponentenbasierten Softwareentwicklung darstellt und in diesem Kapitel zur Validation der Analysemethoden herangezogen wird, ist in Abschnitt 2.2.1 eingeführt.

Die Beispielprozesslandschaft wird sowohl einer statischen als auch einer dynamischen Analyse unterzogen. Allgemein befasst sich die statische Analyse mit der Messung und Bewertung von Attributwerten einer gegebenen Prozesslandschaft ohne ihre direkte Ausführung. Bei der dynamischen Analyse wird die Prozesslandschaft simuliert, um Informationen über Kommunikationseigenschaften wie die durch Kommunikation entstehende Prozessauslastung oder durch das Kommunikationsverhalten entstehende Kosten zu erhalten.

Die Validation der beiden Analysemethoden gibt Auskunft über deren Eignung und Aussagekraft für ihre jeweilige Aufgabenstellung. Zeigler et al. definieren die Validation als „ ... the process of testing a model for validity ... where the validity answers the question of whether it is impossible to distinguish the model and system in the experimental frame of interest“ [ZPK00]. Der Experimentierrahmen spezifiziert dabei die Bedingungen bzw. die inhaltlichen Aspekte, unter denen ein System betrachtet wird. Wird diese Definition auf die Validation der Analysemethoden des Process Landscaping angewandt, müssen die bezüglich der Kommunikation innerhalb einer Prozesslandschaft gemessenen Werte auf ihre Gültigkeit hin überprüft werden. Der Begriff der Gültigkeit umfasst hier insbesondere auch den Nutzen der über die Werte gemessenen und zu bewertenden Eigenschaften der Prozesslandschaft. Derniame et al. beschreiben die Validation als „The process of evaluating the useful properties of a model, by inspection, simulation or test“ [DKW99] und zielen damit ebenfalls auf die Nützlichkeit von Modelleigenschaften ab.

Die Struktur dieses Kapitels folgt den in Abschnitt 2.3 festgelegten Analysezielen, die

sich durch die Untersuchung statischer und dynamischer Kommunikationseigenschaften überprüfen lassen. Für die Analyse statischer Kommunikationseigenschaften wird in Abschnitt 4.1.1 zunächst ein Überblick über die Ausprägung der Eigenschaften der in PLL notierten Beispielprozesslandschaft gegeben. Für diese Ausprägung werden die Kommunikationskostenfaktoren pro Standort und für verschiedene Abstraktionsebenen berechnet. Anschließend wird für unterschiedliche Optimierungsansätze (vgl. Abschnitt 2.3.4) die bezüglich der Kommunikationskosten jeweils günstigste Lösung identifiziert und diskutiert. Die Ergebnisse der so durchgeführten Analyseschritte am konkreten Beispiel werden anschließend unter dem Aspekt ihres allgemeinen Nutzens untersucht.

Für die Analyse dynamischer Kommunikationseigenschaften wird in Abschnitt 4.2 ebenfalls zunächst ein Überblick über die Struktur der Beispielprozesslandschaft PL_{com} und die Ausprägung ihrer Eigenschaften gegeben. Nach der Analyse der Ausgangsbasis in Abschnitt 4.2.2.1 wird in Abschnitt 4.2.2.2 entlang eines konkreten Szenarios ein modifiziertes Simulationsmodell analysiert. Dabei wird überprüft, inwieweit die Modifikationen Auswirkungen auf die Effizienz der Beispielprozesslandschaft haben. Beide Analysedurchläufe betrachten dabei insbesondere die in Abschnitt 2.3.3 definierten Eigenschaften *Kommunikationsaufkommen*, *Prozessauslastung*, *durchschnittliche Pfadlänge* und *Initiator/Responder-Erfüllungsrate*, die die Effizienz einer Prozesslandschaft bezüglich ihres Kommunikationsverhaltens maßgeblich beeinflussen. Abschließend wird in Abschnitt 4.2.3 auch für die Analyse dynamischer Kommunikationseigenschaften eine Nutzenbetrachtung durchgeführt.

Die Basisdaten sowohl für die Analyse statischer als auch dynamischer Kommunikationseigenschaften sind in den Abschnitten 4.1 und 4.2 nicht detailliert aufgeführt. Sie betreffen insbesondere alle Aktivitäten sowie ihre geografische Verteilung innerhalb der Prozesslandschaft, alle Dokumente und existierende Kommunikationskanäle inklusive ihrer Attributierung. Diese und weitere für die Analysen notwendigen Basisdaten finden sich in Anhang A. Er ist unterteilt in Informationen über die Struktur der Prozesslandschaft (Anhang A.1) und über die Wertebelegungen der instanziierten Landschaft sowohl für die Analyse statischer (Anhang A.2) als auch dynamischer Kommunikationseigenschaften (Anhang A.3).

4.1 Analyse statischer Kommunikationseigenschaften

Die Analyse der Kommunikationsstruktur einer Prozesslandschaft $PL_{top-com}$ kann aufgrund fehlender Informationen zum Ablauf und zum Kommunikationsverhalten nur statisch erfolgen. Im zugrundeliegenden Petrinetz wird dazu das Verhältnis zwischen den zur internen und externen Kommunikation genutzten Kommunikationskanälen betrachtet. Die zugehörigen Attribute werden pro Kommunikationskanal ausgewertet.

Im Rahmen des Process Landscaping lässt sich die statische Analyse als die Untersuchung des Aufbaus der Kommunikationsinfrastruktur einer Prozesslandschaft definieren, die anhand vorgegebener Attribute von Kommunikationskanälen und daraus

berechenbaren Kommunikationskostenfaktoren erfolgt. Aufgrund dieses Kkf – und der fehlenden Ablaufinformationen – ist der Abstraktionsgrad eines realen komponentenbasierten Softwareentwicklungsprozesses verglichen mit seiner Darstellung als Prozesslandschaft $PL_{top-com}$ relativ hoch:

- Der Kkf repräsentiert Verhältnisse zwischen internen und externen Kommunikationskosten, er zeigt nicht die absoluten Kommunikationskosten einer Prozesslandschaft auf (vgl. Abschnitt 2.3.4). Je höher also der Kkf, desto ungünstiger ist die Aufteilung der Kosten auf interne und externe Kommunikationskanäle.
- Die Kosten für die Nutzung eines Kommunikationskanals als Bestandteil des Kkf werden abstrakt auf einer Skala von eins bis fünf gemessen: In Abhängigkeit der Belegung der Kommunikationsattribute wird einem Kanal jeweils ein Kostenpunkt zugeordnet, falls

$$\begin{aligned} synch &= 1 \text{ oder } synch = 0 \\ coded &= 1 \\ change &= 0 \\ priv &= 0 \\ mult &= 1 \end{aligned}$$

Eine Analyse auf einem solchen Abstraktionsniveau muss ein konkretes Ziel voraussetzen, um sinnvoll zu sein. Ansonsten sind die resultierenden Zahlenwerte zu abstrakt, um beispielsweise eine korrekte Bewertung des Anstiegs oder der Senkung von Kostenfaktoren durchführen zu können. Daher dient die nachfolgende Diskussion der Ausgangsbasis in Abschnitt 4.1.1 zunächst dem Verständnis der Analyseschritte und der Erläuterung der wichtigsten berechneten Werte. In Abschnitt 4.1.2.2 werden diese Werte zur vergleichenden Analyse herangezogen, um bezüglich verschiedener Optimierungsziele die beste Lösung zu identifizieren.

4.1.1 Struktur der Prozesslandschaft $\mathrm{PL}_{top-com}$

Die in Abschnitt 2.2.1 eingeführte Beispielprozesslandschaft ist nach logischen Gesichtspunkten modelliert. Um bei der Analyse auch die geografische Verteilung der Aktivitäten zu berücksichtigen, ist die Prozesslandschaft zunächst in ihre lokale Sicht umstrukturiert worden. Diese Umstrukturierung ist gemäß der in Abschnitt 3.2.2.1 formulierten Definition 3.2.44 und Konsistenzbedingung 3.2.46 erfolgt: Während die Definition 3.2.44 Anzahl und Struktur der Aktivitäten in einer lokalen Sicht festlegt, beschreibt die Konsistenzbedingung 3.2.46 die oberen Grenzen für die Anzahl der Schnittstellen, die sich durch die Umstrukturierung aus der logischen Sicht ergibt.

Abbildung 4.1 zeigt die lokale Sicht auf den Aktivitätenbaum der Prozesslandschaft. Diese bildet den Ausgangspunkt der Analyse. Waren in der logischen Sicht noch drei

Abstraktionsebenen modelliert, ist die lokale Sicht zunächst auf zwei Abstraktionsebenen beschränkt worden.

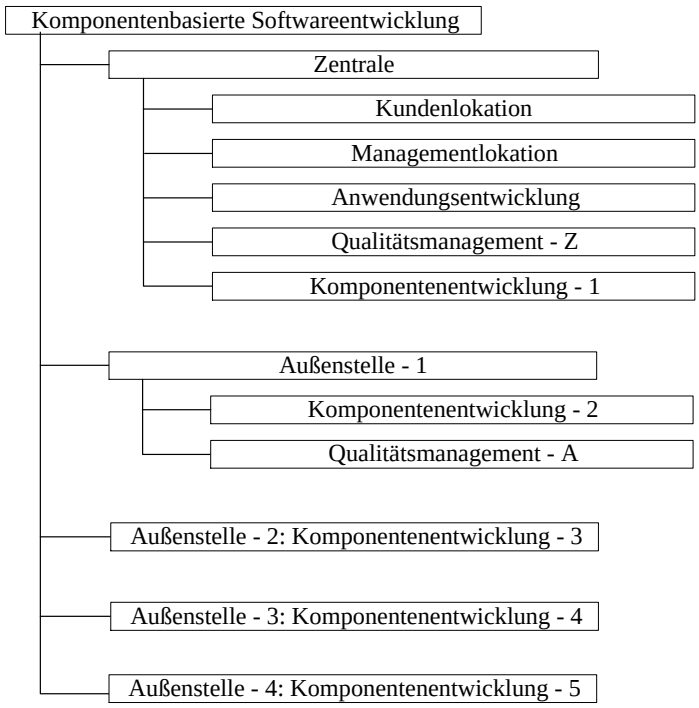


Abbildung 4.1: Lokale Sicht der Beispielprozesslandschaft $PL_{top-com}$

Auf der obersten Ebene der lokalen Sicht existieren insgesamt fünf Orte, eine Zentrale und vier Außenstellen. Die Zentrale ist in je einen Standort für das Application Engineering (Standort *Anwendungsentwicklung*), das Projektmanagement inklusive des Domain Engineering (Standort *Managementlokation*), ein Qualitätsmanagement (Standort *Qualitätsmanagement-Z*) und einen Standort für das Component Engineering (Standort *Komponentenentwicklung-1*) unterteilt. Zusätzlich existiert hier eine Kundenlokation, an der einige Aktivitäten ausgeführt werden, die intensive Zusammenarbeit mit dem Kunden erfordern. An allen Außenstellen werden Aktivitäten zur Entwicklung weiterer Komponenten durchgeführt (*Komponentenentwicklung-2* bis *Komponentenentwicklung-5*). Außenstelle-1 weist zusätzlich noch eigene Qualitätsmanagement-Aktivitäten auf (Standort *Qualitätsmanagement-A*). Für alle anderen Außenstellen übernimmt das Qualitätsmanagement der Zentrale die Test- und Reviewaktivitäten.

Tabelle 4.1 gibt einen Überblick über die Komplexität der Beispielprozesslandschaft. Durch die Verteilung einiger Aktivitäten auf mehrere Standorte hat sich beim Wechsel von der logischen zur lokalen Sicht die Gesamtzahl der zu betrachtenden Aktivitäten erhöht. Des Weiteren sind den verschiedenen Datenflüssen insgesamt 269

(99 plus 170) Kommunikationskanäle zur internen und externen Kommunikation zugeordnet. Diese sind jeweils mit Kommunikationsattributen versehen und für die Analyse mit 3228 (zwölf Werte pro Kommunikationskanal) konkreten Werten belegt. Alle Kommunikationskanäle wurden so instanziiert, dass sie funktionsfähig sind (vgl. Abschnitt 3.1.3). Für eine detaillierte Auflistung aller Aktivitäten, ihrer Verteilung sowie bestehender Kommunikationskanäle sei auf Anhang A.1 verwiesen. Im Anhang A.2 sind die genauen Werte aller der nachfolgenden Analyse zugrundeliegenden Kommunikationsattribute aufgeführt.

	Anzahl
Dokumente	69
Aktivitäten	114
Abstraktionsebenen	2
Orte auf oberster Ebene	5
Orte auf unterer Ebene	10
Kommunikationskanäle zur externen Kommunikation	99
Kommunikationskanäle zur internen Kommunikation	170
Attribute pro Kommunikationskanal	12
Wertzuweisungen an Aktivitäten	114
Wertzuweisungen an Kommunikationskanäle	3228

Tabelle 4.1: Aufbau der Prozesslandschaft

4.1.2 Analyse der Kommunikationsinfrastruktur einer Prozesslandschaft $\mathbf{PL}_{top-com}$

4.1.2.1 Analyse der Ausgangsbasis

Für jeden der zehn Standorte der Beispielprozesslandschaft sind die nachfolgend aufgeführten Eigenschaften gemäß den in den Abschnitten 2.3.4 und 3.1.3 eingeführten Definitionen gemessen bzw. berechnet worden:

- Kopplungsdichte zwischen zwei ausgewählten Orten
- Kopplungsdichte pro Ort (Kopplungsdichte bzgl. aller mit diesem kommunizierenden Ort)
- Kopplungskomplexität zwischen Orten
- Kopplungskomplexität pro Ort (Kopplungskomplexität bzgl. aller mit diesem kommunizierenden Ort)
- Kopplungszahl
- Kohäsionsdichte pro Ort

- Kohäsionskomplexität pro Ort
- Kohäsionszahl

Die einzelnen Ergebnisse sind in den Tabellen 4.2 bis 4.5 aufgeführt. Sie bilden die Grundlage zur Berechnung der Kommunikationskostenfaktoren Kkf, die unabhängig vom konkret verfolgten Optimierungsansatz minimal gehalten werden sollen, um eine mit Blick auf die statischen Kommunikationseigenschaften möglichst effiziente Prozesslandschaft zu erhalten.

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation	6	5
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung	Managementlokation	13	7
Anwendungsentwicklung	Qualitätsmanagement-Z	12	5
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle 4.2: Kopplung zwischen zwei Orten

Bei den berechneten Werten für die Kopplungsdichte zwischen zwei Orten (vgl. Tabelle 4.2) fallen diejenigen zwischen der Anwendungsentwicklung und der Managementlokation und zwischen der Anwendungsentwicklung und dem zentralen Qualitätsmanagement auf, da hier die Werte deutlich höher als bei anderen Kopplungsdichten liegen. Führt man sich aber die Aktivitäten an den Standorten *Managementlokation* und

Qualitätsmanagement-Z vor Augen, werden die höheren Werte verständlich, da diese vor allem Kontroll- und Steuerungsaufgaben für die gesamte Softwareentwicklung umfassen.

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	12
Managementlokation	54	12
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle 4.3: Kopplung eines Ortes

Die in Tabelle 4.3 aufgeführte Kopplungsdichte der Orte *Managementlokation*, *Qualitätsmanagement-Z* und *Anwendungsentwicklung* ist erwartungsgemäß ebenfalls höher. Sie bezieht sich auf die Anzahl aller Aktivitäten an anderen Orten, die mit Aktivitäten am betrachteten Ort kommunizieren. Die Werte der Kopplungskomplexitäten für die Kommunikationskanäle pro Standort (zu allen anderen Standorten der Prozesslandschaft) liegen meist knapp unter der Summe ihrer Kopplungskomplexitäten zu einzelnen anderen Standorten (vgl. Tabelle 4.2). Bei der Berechnung dieser Summe ist zu beachten, dass mehrfach existierende Ausprägungen eines Kommunikationskanals nur einmal gezählt werden. Das Ergebnis lässt darauf schließen, dass Aktivitäten, die innerhalb der Zentrale ausgeführt werden, für die Kommunikation zur Außenstelle-1 ähnlich ausgeprägte Kanäle wie zu den übrigen Außenstellen verwenden, ansonsten jedoch eine relativ heterogene Kommunikationsinfrastruktur vorliegt.

Zur Berechnung der verschiedenen Kommunikationskostenfaktoren Kkf müssen auch die Kohäsionsdichten und -komplexitäten der einzelnen Standorte bestimmt werden. Diese sind in Tabelle 4.4 aufgeführt.

Bei einem Vergleich der Kohäsionsdichten zu den in Tabelle 4.2 aufgeführten Kopplungsdichten fällt auf, dass diese bei den Standorten *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* deutlich höher liegen (vgl. Tabelle 4.4). Dies lässt sich jedoch durch die Aufgabenstellung der dort durchgeführten Aktivitäten erklären. Dort werden die einzelnen Komponenten des zu erstellenden Gesamtsystems entwickelt. Die Kommunikation zur Zentrale, die die Kopplungsdichte beeinflusst, erfolgt daher hauptsächlich aufgrund von Prüfungs- und Steuerungsaktivitäten sowie von Auslieferungen der entwickelten Komponenten. Ansonsten werden die Aufgaben losgelöst vom Kernprozess des Application Engineering, aber mit verstärkter interner Kommunikation, durchgeführt. Letzteres führt wiederum zu den vergleichsweise hohen Kopplungsdichten.

Ort	Kohäsionsdichte	Kohäsionskomplexität
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation	19	12
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle 4.4: Kohäsion eines Ortes

Die Komplexitäten der Kommunikationskanäle sind bei interner und externer Kommunikation der einzelnen Standorte jeweils ähnlich. Hier fällt die vergleichsweise hohe Komplexität sowohl der Kopplung als auch der Kohäsion des Standortes *Managementlokation* auf. Dieser Standort kommuniziert jedoch als einziger mit allen anderen Standorten der Prozesslandschaft (der Standort *Anwendungsentwicklung* weist keinen Kommunikationskanal zum Standort *Qualitätsmanagement-A* auf) und beinhaltet außerdem zwei aus logischer Sicht unterschiedliche Aufgabenbereiche (Projektmanagement und Domain Engineering), die jeweils unterschiedliche Anforderungen an die Kommunikationsinfrastruktur haben. Beide Aspekte zusammen verursachen die in den Tabellen 4.3 und 4.4 aufgeführten erhöhten Komplexitätswerte.

Ort	Anzahl Aktivitäten	Kopplungszahl	Kohäsionszahl
Kundenlokation	3	3	2
Anwendungsentwicklung	14	13	14
Managementlokation	16	12	13
Komponentenentwicklung-1	13	8	12
Komponentenentwicklung-2	13	8	12
Komponentenentwicklung-3	13	8	12
Komponentenentwicklung-4	13	8	12
Komponentenentwicklung-5	13	8	12
Qualitätsmanagement-Z	8	6	8
Qualitätsmanagement-A	8	6	8

Tabelle 4.5: Kopplungs- und Kohäsionszahl eines Ortes

Die berechneten Kopplungs- und Kohäsionszahlen in Tabelle 4.5 zeigen auf, wie viele der Aktivitäten pro Standort an interner bzw. an externer Kommunikation beteiligt sind. Die Aktivitäten am Standort *Anwendungsentwicklung* sind fast vollständig

sowohl an interner als auch an externer Kommunikation beteiligt. Die Aktivitäten der Standorte *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* sind dagegen hauptsächlich an interner Kommunikation beteiligt. Diese Verschiebung lässt sich ebenfalls mit den Aufgaben der an den jeweiligen Standorten durchgeführten Aktivitäten erklären. Damit erweisen sich die Kopplungs- und Kohäsionszahlen insbesondere für den Vergleich verschiedener Situationen und ihrer Interpretation als hilfreich.

Die bislang erörterten Kopplungs- und Kohäsionswerte der Beispielprozesslandschaft $PL_{top-com}$ machen den ersten Teil der für eine Analyse der Kommunikationskosten notwendigen Daten aus. Der zweite Teil beinhaltet Informationen über das eingesetzte Kostenmodell. Da auf den betrachteten Ebenen der Prozesslandschaft keine Ablaufinformationen berücksichtigt worden sind, wird im Rahmen der Analyse die Nutzungshäufigkeit jedes Kommunikationskanals abstrakt auf einer Skala von eins bis zehn festgelegt (zur Motivation vgl. Abschnitt 2.3.4). Auch die Kosten pro übertragener Informationseinheit werden für jeden Kanal auf einer Skala von eins bis fünf bestimmt. Diese ergibt sich aus der Anzahl möglicher Kostenpunkte, die einem Kommunikationskanal in Abhängigkeit der Ausprägung seiner Attribute zugeordnet worden sind (vgl. Abschnitt 4.1).

Insgesamt ist es mit dem so aufgebauten Kkf

$$\frac{Kopplungsdichte(o) \times (Nutzungsh\ddot{a}ufigkeit \times Kosten)\ pro\ externer\ Kanal}{Koh\ddot{a}sionsdichte(o) \times (Nutzungsh\ddot{a}ufigkeit \times Kosten)\ pro\ interner\ Kanal}$$

möglich, auf einer einfachen Bemessungsgrundlage für die statischen Kommunikationseigenschaften einer Prozesslandschaft zu bleiben. Für die konkreten Werte der Nutzungshäufigkeit pro Kommunikationskanal und Kosten pro Informationseinheit sei wieder auf Anhang A.2 verwiesen. Nachfolgend sind in Tabelle 4.6 die berechneten Kommunikationskostenfaktoren für jeden Standort aufgeführt.

Ort	**Kkf**
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation	6,14
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle 4.6: Kommunikationskostenfaktoren aller Orte

Der Kkf der Kundenlokation fällt hier deutlich aus dem Rahmen. Die interne Kommunikation an diesem Ort ist nur unvollständig modelliert, da sie für den Software-

entwicklungsprozess kaum eine Rolle spielt. Beim Kunden selbst wird das Anforderungsdokument erstellt und die Abnahme des Softwareproduktes durchgeführt. Lediglich die externe Kommunikation spielt eine Rolle, da sie unter anderem den Softwareentwicklungsprozess mit ersten Daten in Form des Anforderungsdokumentes beliefert. Da jedoch kein Softwareentwicklungsprojekt Einfluss auf die Kundenlokation hat, bleibt dieser Standort im Verlauf der Analyse weitestgehend unberührt.

In Tabelle 4.6 fallen weiter die erhöhten Kommunikationskostenfaktoren der Standorte *Anwendungsentwicklung* und *Managementlokation* auf. Diese sind nicht durch die Anzahl der dortigen Aktivitäten zu erklären, sondern vor allem durch die Anzahl der jeweils an externer Kommunikation beteiligten Aktivitäten, die deutlich von der an anderen Standorten abweicht (vgl. Kopplungszahlen in Tabelle 4.5). Am Standort *Anwendungsentwicklung* wirkt sich zusätzlich noch die erhöhte Komplexität der externen Kommunikationskanäle aus, die die Kosten ihrer Nutzung beeinflusst. Eine erhöhte Komplexität der internen Kommunikationskanäle wie beim Standort *Managementlokation* wirkt sich dagegen nicht so stark aus, da interne Kommunikation im Verhältnis zur externen Kommunikation immer als preisgünstiger angesehen wird.

Mit der bislang durchgeführten Diskussion ist ein Gesamtbild über die statische Kommunikationsinfrastruktur und ihrer Kosten einer mit konkreten Werten instanziierten Beispielprozesslandschaft gegeben. Es dient als Ausgangsbasis für vergleichende Analysen mit modifizierten Prozesslandschaften.

4.1.2.2 Analyse verschiedener Optimierungsansätze

Im Folgenden werden die in Abschnitt 2.3.4 aufgeführten Strategien zur strukturellen Veränderung einer Prozesslandschaft durchgespielt. Die Anwendung dieser Strategien führt zu verschiedenen Landschaftsvarianten mit unterschiedlichen Kommunikationsstrukturen, deren Kostenfaktoren miteinander verglichen werden, um die jeweils bezüglich der Kommunikationskosten günstigste Lösung zu identifizieren. Die Strategien sind im Einzelnen:

- Aufbau eines neuen Standortes
- Zusammenlegen mehrerer Orte
- Verlagerung von Aktivitäten
- Verlagerung von Verantwortlichkeiten bei mehrfach vorhandenen Aktivitäten

Für jede Strategie werden konkrete Szenarien entwickelt und die Ergebnisse des Optimierungsansatzes – die Minimierung des jeweiligen Kkf – diskutiert. So können über den Vergleich verschiedener Kostenfaktoren sinnvolle Änderungen identifiziert werden.

Aufbau eines neuen Standortes

Szenario:

Im Rahmen dieses Szenarios wird innerhalb der Zentrale ein zusätzlicher Standort errichtet. An diesem können beispielsweise Teilaktivitäten der Managementlokation durchgeführt werden, an der bislang sowohl Projektmanagementaktivitäten als auch Aktivitäten aus dem Bereich des Domain Engineering ausgeführt werden. Letztere sollen zukünftig am zusätzlichen Standort stattfinden. Alternativ könnten aber auch Aktivitäten aus dem Konfigurationsmanagement des Application Engineering (Standort *Anwendungsentwicklung*) oder die Tests und Reviews des Qualitätsmanagements (Standort *Qualitätsmanagement-Z*) ausgelagert werden. Die Berechnung des Kkf für alle drei Alternativen wird zeigen, welche die kostengünstigste ist.

Zur Vereinfachung wird die Auslagerung des Domain Engineering (drei Aktivitäten) zu einem neuen Standort in Tabelle 4.7 als *Alternative1* bezeichnet, die Auslagerung des Konfigurationsmanagements (drei Aktivitäten) vom Standort *Anwendungsentwicklung* als *Alternative2* und die Auslagerung der Test- und Reviewtätigkeiten (fünf Aktivitäten) des zentralen Qualitätsmanagements als *Alternative3*.

Ort	**Kkf** Ausgangsbasis	**Kkf** Alternative1	**Kkf** Alternative2	**Kkf** Alternative3
Kundenlokation	186,33			
Anwendungsentwicklung	8,87		10,94	
Managementlokation	6,14	7,72		
Komponentenentwicklung-1	0,26			
Komponentenentwicklung-2	0,26			
Komponentenentwicklung-3	0,26			
Komponentenentwicklung-4	0,26			
Komponentenentwicklung-5	0,26			
Qualitätsmanagement-Z	2,28			1241,66
Qualitätsmanagement-A	0,15			
neuer Standort		23,55	18,7	1209,6

Tabelle 4.7: Kommunikationskostenfaktoren aller Orte – Alternativen 1 bis 3

Tabelle 4.7 zeigt die Auswirkungen der drei Alternativen auf die Kommunikationskostenfaktoren der jeweils betroffenen Standorte. Die dazu notwendigen Neuberechnungen der Kopplungs- und Kohäsionsdichten sind im Anhang A.2 in den Tabellen A.19 bis A.30 detailliert dargestellt. Bei der Auslagerung von Aktivitäten zum neuen Standort ist jeweils darauf geachtet worden, logisch eng zusammengehörende Bereiche nicht voneinander zu trennen. So ist bei *Alternative1* der gesamte Bereich des Domain Engineering ausgelagert worden, bei *Alternative2* die vollständig zum Konfigurationsmanagement des Application Engineering gehörende Menge an Aktivitäten und bei *Alternative3* ausschließlich diejenigen Bereiche des Qualitätsmanagements, die zur operativen Qualitätssicherung zählen.

Trotzdem sind bei den drei Alternativen große Unterschiede in den Auswirkungen auf die betroffenen Kommunikationskostenfaktoren festzustellen. Insbesondere eine Aufteilung des zentralen Qualitätsmanagements auf zwei Standorte (Alternative3) hätte negative Auswirkungen auf den Kkf sowohl des alten als auch des neuen Standortes. Dabei schien sich das zentrale Qualitätsmanagement – aufgrund der im Vergleich zu den Außenstellen ebenfalls relativ hohen Kommunikationskosten – für eine Aufteilung genauso anzubieten wie die Standorte *Anwendungsentwicklung* und *Managementlokation*. Der Grund liegt in der relativ geringen Kohäsion der beiden betrachteten Orte: Die am Standort *Qualitätsmanagement-Z* verbleibenden Aktivitäten nutzen beispielsweise nur noch genau einen Kommunikationskanal zur internen Kommunikation (siehe Tabelle A.29 in Anhang A.2). Am neuen Standort werden zwar mehr interne Kommunikationskanäle genutzt, die externen werden jedoch um ein Vielfaches häufiger genutzt, da über sie der Informationsaustausch bezüglich der Reviews und Tests von insgesamt vier an verschiedenen Standorten zu entwickelnden Komponenten verläuft. Da der Kkf über das Verhältnis zwischen internen und externen Kommunikationsfaktoren berechnet wird, verschlechtert sich dieses kostenungünstige Verhältnis noch weiter und resultiert in den in Tabelle 4.7 aufgeführten Ergebnissen. Dass diese für die Alternative3 jedoch so negativ ausfallen würden, ist ein Analyseergebnis, welches durch „bloßes Hinsehen“ nicht erkennbar gewesen ist.

Auch die Aufteilung der Managementlokation (*Alternative1*) in ihre logischen Bestandteile *Projektmanagement* und *Domain Engineering* ist nicht die kostengünstigste Alternative für die Errichtung eines neuen Standortes. Bei dieser hätte die Managementlokation eine Kostensteigerung von mehr als 25% zu erwarten. Bezüglich des Kkf hat sich die Verlagerung des Konfigurationsmanagements aus dem Application Engineering als die aus Kostensicht beste Lösung herausgestellt. Die Kostensteigerung für den verkleinerten Standort *Anwendungsentwicklung* beträgt hier $23,34\%$. Aber auch die Gesamtkosten, die bei der Errichtung eines zusätzlichen Standortes entstehen, sind hier mit einem Kkf von insgesamt $29,64$ ($10,94$ plus $18,7$) die günstigsten aller drei diskutierten Alternativen.

Nach der Identifikation der zahlenmäßig günstigsten Alternative sollte diese immer auch auf die Auswirkung ihrer Umsetzung untersucht werden. Im vorliegenden Fall der Aufteilung des Application Engineering und seines Konfigurationsmanagements auf verschiedene Standorte hätte die Umsetzung beispielsweise eine in der Realität eher unübliche Arbeitssituation zur Folge, da ein Softwareentwicklungsteam zumeist gleichzeitig für Konfigurationsmanagementaufgaben verantwortlich ist. Ein Grund für diese Aufgabenzuordnung ist das Wissen um die Softwarearchitektur, das sowohl für die Entwicklung als auch für die Konfiguration eines Softwaresystems erforderlich ist. Eine geografische Aufteilung der Aktivitäten hätte hier wahrscheinlich negative Auswirkungen auf den Arbeitsablauf, da zuvor zentriertes Wissen ebenfalls aufgeteilt würde.

Bei *Alternative1* ist die Situation anders gelagert. Die Auskopplung des Domain Engineering vom Managementstandort, die mit einem Kkf von insgesamt $31,27$ ($7,72$ plus $23,55$) geringfügig teurer ist als die Aufteilung des Application Engineering

und seines Konfigurationsmanagements auf verschiedene Standorte, erscheint aus semantischer Sicht sinnvoller: Das Domain Engineering beinhaltet vor allem operative Aktivitäten, während das Projektmanagement Steuerungs- und Kontrollaufgaben durchführt. Eine Trennung dieser beiden Aufgabenbereiche hätte keine Aufteilung von für alle Aufgaben wichtigen Wissensträger auf verschiedene Standorte zur Folge. Da die Steigerung der Kommunikationskosten vergleichsweise ähnlich ist, sollte daher *Alternative1* für die Umsetzung ausgewählt werden.

Zusammenlegen mehrerer Orte

Szenario:

Für das Zusammenlegen mehrerer Orte wird die kostengünstigste aus allen Kombinationen von jeweils zwei Orten innerhalb der Zentrale berechnet. Lediglich die Kundenlokation bleibt unberücksichtigt. Die existierenden Außenstellen werden ebenfalls nicht in die Überlegungen dieses Szenarios einbezogen.

Für die Zentrale existieren insgesamt sechs verschiedene Möglichkeiten, jeweils zwei Standorte zusammenzulegen. Tabelle 4.8 zeigt die Auswirkungen auf den Kkf durch ein Zusammenlegen verschiedener Orte. Für ihre detaillierte Berechnung sei wieder auf Anhang A.2, Tabellen A.31 bis A.54, verwiesen. Innerhalb der Tabellen werden die unterschiedlichen Möglichkeiten der Zusammenlegung als *Fusionsalternative1* bis *Fusionsalternative6* bezeichnet. Die Sternchen in Tabelle 4.8 kennzeichnen die jeweils zusammengelegten Standorte und erleichtern den Vergleich mit den Kommunikationskostenfaktoren der Ausgangsbasis.

Das Zusammenlegen zweier Standorte kommt dem Bestreben nach einer möglichst geringen Kopplung und einer möglichst hohen Kohäsion entgegen, da zur externen Kommunikation verwendete Kommunikationskanäle hierbei teilweise zu internen werden. Bezogen auf die Beispielprozesslandschaft bedeutet damit jede Fusionsalternative eine Verringerung der Kommunikationskosten. Für die Identifikation der kostengünstigsten Zusammenlegung zweier Standorte hat sich gezeigt, dass die Werte der Ausgangsbasis ausreichend sind. Eine Neuberechnung für alle Fusionsalternativen ist nur für den Fall erforderlich, dass die konkreten Kommunikationskostenfaktoren der fusionierten Standorte von Interesse sind.

Betrachtet man die für die jeweiligen Fusionsalternativen 1 bis 6 berechneten Kommunikationskostenfaktoren, könnte man zunächst glauben, dass *Fusionsalternative3* mit dem geringsten Kkf das Maximum an Kostenersparnis bringen wird. Hier ist der Kkf von zuvor ingesamt 11,15 (8,87 plus 2,28) auf 0,57 gesunken. Tatsächlich ist aber das Verhältnis der beiden alten zum jeweils neuen Kkf zu betrachten. Bei *Fusionsalternative3* macht dies eine Differenz von 10,58 aus. Demnach wird die Zusammenlegung der beiden Standorte *Anwendungsentwicklung* und *Managementlokation* mit 13,36 (*Fusionsalternative1*) das Maximum an Kostenersparnis erreichen.

Ausschlaggebend für die Zusammenlegung der Standorte *Anwendungsentwicklung* und *Managementlokation* als kostengünstigste Alternative ist, dass am zusammengelegten Standort die Mehrheit aller am Softwareentwicklungsprozess beteiligten Aktivitäten stattfindet und gleichzeitig die beiden Standorte mit den zuvor schon höchsten

Ort	**Kkf** Ausgangsbasis	**Kkf** Alt.1	**Kkf** Alt.2	**Kkf** Alt.3	**Kkf** Alt.4	**Kkf** Alt.5	**Kkf** Alt.6
Kundenlokation	186,33						
Anwendungsentwicklung	8,87	*	*	*			
Managementlokation	6,14	*			*	*	
Komponentenentwicklung-1	0,26		*		*		*
Komponentenentwicklung-2	0,26						
Komponentenentwicklung-3	0,26						
Komponentenentwicklung-4	0,26						
Komponentenentwicklung-5	0,26						
Qualitätsmanagement-Z	2,28			*		*	*
Qualitätsmanagement-A	0,15						
Anwendungsentwicklung inkl. Managementlokation		1,65					
Anwendungsentwicklung inkl. Komponentenentwicklung-1			1,51				
Anwendungsentwicklung inkl. Qualitätsmanagement-Z				0,57			
Managementlokation inkl. Komponentenentwicklung-1					1,12		
Managementlokation inkl. Qualitätsmanagement-Z						3,61	
Qualitätsmanagement-Z inkl. Komponentenentwicklung-1							0,77

Tabelle 4.8: Kommunikationskostenfaktoren aller Orte – Fusionsalternativen 1 bis 6

Kommunikationskostenfaktoren zusammengelegt werden. Dies entspräche jedoch einem Optimierungsansatz, der – wie in Abschnitt 2.3.4 bereits angesprochen – ausschließlich den Kkf berücksichtigt und daher nicht immer nutzbringend ist. Für das Zusammenlegen von Standorten ist das rechnerisch günstigste Ergebnis deshalb unbedingt, wie zuvor beim Aufbau eines neuen Standortes, auf die Auswirkung seiner Umsetzung zu untersuchen.

Die zuvor erwähnte Fusionsalternative3 bietet die zweitgünstigste Kostenersparnis. Sie sollte ebenfalls für eine Umsetzung in Betracht gezogen werden, da hier zwei Aufgabenbereiche zusammengelegt würden, die viele Berührungspunkte haben: Jedes Teilergebnis des Application Engineering muss vom Qualitätsmanagement überprüft werden, bevor es weiter verarbeitet wird.

Es zeigt sich, dass gerade für die Zusammenlegung von Standorten ein Kommunikationskostenfaktor nur eines von mehreren Entscheidungskriterien sein sollte. Der Kkf kann jedoch bei der Entscheidungsfindung als neutrales Argument sinnvoll eingesetzt werden, wenn mehrere Alternativen aufgrund anderer Kriterien vorliegen.

Verlagerung von Aktivitäten

Szenario:

Die Aktivität *Anforderungsanalyse erstellen* soll von der Kundenlokation an einen anderen Ort der Zentrale verlagert werden. Auch hierfür wird die kostengünstigste Alternative berechnet.

Die bestmögliche Alternative für die Verlagerung einer einzelnen – wenn auch komplexen – Aktivität ist meist eine mit starker Kopplung an Aktivitäten, die an anderen Standorten stattfinden. Hintergrund dieses Optimierungsansatzes ist damit die Absicht nach lokaler Verbesserung eines Kkf, jedoch mit Blick auf die Auswirkung auf die Kommunikationskosten für die Prozesslandschaft insgesamt.

Ort	**Kkf** Ausgangsbasis	**Kkf** Alt.1	**Kkf** Alt.2	**Kkf** Alt.3	**Kkf** Alt.4
Kundenlokation	186,33	32	32	32	32
Anwendungsentwicklung	8,87	4,05			
Managementlokation	6,14		6,45		
Komponentenentwicklung-1	0,26				0,97
Komponentenentwicklung-2	0,26				
Komponentenentwicklung-3	0,26				
Komponentenentwicklung-4	0,26				
Komponentenentwicklung-5	0,26				
Qualitätsmanagement-Z	2,28			1,98	
Qualitätsmanagement-A	0,15				

Tabelle 4.9: Kommunikationskostenfaktoren aller Orte – Verlagerungsalternativen 1 bis 4

Tabelle 4.9 zeigt die für die Verlagerung der Aktivität *Anforderungsanalyse erstellen* möglichen Alternativen innerhalb der Zentrale und die jeweils damit verbundenen veränderten Kommunikationskostenfaktoren. Die detaillierten Berechnungsgrundlagen finden sich in den Tabellen A.55 bis A.70 in Anhang A.2.

Die drastische Verringerung der Kommunikationskosten an der Kundenlokation erstaunt nicht weiter, da die dort entfernte Aktivität nur einen Kommunikationskanal für die interne Kommunikation verwendet hat, aber acht zur externen Kommunikation (vgl. Tabellen A.2 bis A.4 in Anhang A.2). Zusätzlich war die Kommunikation zum zentralen Qualitätsmanagement aufgrund der Nutzungshäufigkeit und der Kosten pro Informationseinheit vergleichsweise teuer (vgl. Tabelle A.3 in Anhang A.2). Durch die Verlagerung ist die Kundenlokation jetzt von diesen Kosten befreit.

Es fällt auf, dass bei zwei Verlagerungsalternativen (1 und 3) nicht nur die Gesamtkosten der Prozesslandschaft gesenkt werden können, sondern gleichzeitig auch die lokalen Kommunikationskosten derjenigen Standorte, die alternativ um die Aktivität *Anforderungsanalyse erstellen* erweitert worden sind. Am Standort *Anwendungsentwicklung* konnten diese sogar um mehr als 50% gesenkt werden. *Verlagerungsalternative1* ist damit als die kostengünstigste identifiziert. Ihre Umsetzung ist auch für den Ablauf der Softwareentwicklung sinnvoll, da das Ergebnis der Aktivität *Anforderungsanalyse erstellen* in vielen Bereichen des Application Engineering benötigt wird und seine Erstellung am gleichen Standort einen einfachen und schnellen Zugriff ermöglicht.

Des Weiteren kann festgehalten werden, dass sich lokale Verbesserungen eines Kkf auch positiv auf andere Standorte innerhalb einer Prozesslandschaft auswirken können. Der erwartete Nutzen der für die Identifikation der kostengünstigsten Alternative durchzuführenden Berechnungen hat sich damit noch weiter erhöht.

Verlagerung von Verantwortlichkeiten bei mehrfach vorhandenen Aktivitäten

Szenario:

Das Qualitätsmanagement des zentralen Standortes weist, verglichen mit dem Qualitätsmanagement der Außenstelle-1, ein wesentlich höheres Arbeits- und Kommunikationsaufkommen auf. In der Zentrale werden die Testaktivitäten für vier von fünf Komponentenentwicklungsstandorten und zusätzlich noch die Reviews und Tests für die Entwicklung des Gesamtsystems durchgeführt. Das Qualitätsmanagement der Außenstelle-1 wickelt dagegen lediglich die Testaktivitäten des Standortes *Komponentenentwicklung-2* ab, ein vergleichsweise geringer Arbeits- und Kommunikationsaufwand. Trotzdem müssen hier bis auf drei Aktivitäten alle Qualitätsmanagementaktivitäten zur Verfügung stehen. Hier soll nun analysiert werden, wie sich die Durchführung der Testaktivitäten von vier Komponentenentwicklungsstandorten durch das Qualitätsmanagement der Außenstelle-1 auf den Kkf der Prozesslandschaft auswirkt. Das zentrale Qualitätsmanagement soll dann nur noch für das Application Engineering und das Component Engineering der Zentrale verantwortlich sein, während das Qualitätsmanagement der Außenstelle-1 sich um alle außerhalb der Zentrale entwickelten Komponenten kümmert.

Durch die Verschiebung der Verantwortlichkeiten verändern sich lediglich die Kommunikationskostenfaktoren der beiden Qualitätsmanagementstandorte, und dies auch nur aufgrund veränderter Kopplungsdichten. Die jeweiligen Kohäsionsdichten und -komplexitäten sowie die dadurch entstehenden internen Kommunikationskosten verändern sich nicht. Neben der Verschiebung der Kopplungsdichten verändert sich des Weiteren noch die Nutzungshäufigkeit bestehender externer Kanäle, da sich zwar der Auslastungsgrad der kommunizierenden Aktivitäten geändert hat, aber nicht ihre Anzahl. Die Standorte *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* weisen einen unveränderten Kkf auf. Ein Teil der externen Kommunikation der Standorte *Komponentenentwicklung-3* bis *Komponentenentwicklung-5* verläuft jetzt über den Standort *Qualitätsmanagement-A* statt über *Qualitätsmanagement-Z*. Die veränderten Kommunikationskostenfaktoren dieser beiden Qualitätsmanagementstandorte sind in Tabelle 4.10 zusammen mit denen der Ausgangsbasis dargestellt.

Ort	**Kkf** Ausgangsbasis	**Kkf** nach Verschiebung
Kundenlokation	186,33	
Anwendungsentwicklung	8,87	
Managementlokation	6,14	
Qualitätsmanagement-Z	2,28	1,12
Qualitätsmanagement-A	0,15	1,18
Komponentenentwicklung-1	0,26	
Komponentenentwicklung-2	0,26	
Komponentenentwicklung-3	0,26	
Komponentenentwicklung-4	0,26	
Komponentenentwicklung-5	0,26	

Tabelle 4.10: Kommunikationskostenfaktoren aller Orte nach Verschiebung der QM-Verantwortlichkeiten

Während die Kosten für den Standort *Qualitätsmanagement-Z* um über 50% gesunken sind, steigen die Kosten für den Standort *Qualitätsmanagement-A* verhältnismäßig stark an. Trotzdem liegt der Wert bei einem Vergleich der Kommunikationskostenfaktoren aller Standorte (ausgenommen *Komponentenentwicklung-1* bis *Komponentenentwicklung-5*) mit $1,18$ noch an zweitgünstigster Stelle. Auch die Gesamtkosten der Prozesslandschaft sind um den Wert $0,13$ leicht gesunken.

Betrachtet man die Kommunikationskostenfaktoren der obersten Abstraktionsebene nach Verschiebung der Verantwortlichkeiten (vgl. Tabelle 4.11), hat die Außenstelle-1 jetzt den höchsten Kkf. Dieser ist fast auf das ursprüngliche Niveau des Kkf der Zentrale angestiegen. Bedenkt man, dass an der Zentrale die größte Anzahl aller am Softwareentwicklungsprozess teilnehmenden Aktivitäten stattfindet, erscheint der Kkf der Außenstelle-1 jetzt unverhältnismäßig hoch.

Ort	**Kkf** Ausgangsbasis	**Kkf** nach Verschiebung
Zentrale	0,30	0,04
Außenstelle-1	0,09	0,29
Außenstelle-2	0,26	
Außenstelle-3	0,26	
Außenstelle-4	0,26	

Tabelle 4.11: Kommunikationskostenfaktoren aller Orte auf oberster Ebene nach Verschiebung der QM-Verantwortlichkeiten

An dieser Stelle stößt die Analyse der Kommunikationskosten einer Prozesslandschaft allein auf der Basis statischer Kommunikationsstrukturen und abstrakter Skalen für die Nutzungshäufigkeit pro Kommunikationskanal sowie Kosten pro Informationseinheit an ihre Grenzen. Weitere Analysen sind für eine solche Situation nur unter Berücksichtigung zusätzlicher, dynamischer Kommunikationseigenschaften hilfreich. Tendenziell scheint eine Verschiebung der Verantwortlichkeiten, wie im Szenario vorgeschlagen, sinnvoll. Wie groß dieser Nutzen aber tatsächlich sowohl in Bezug auf die Kommunikationskosten als auch auf den Auslastungsgrad ist, lässt sich hier nicht mehr näher bestimmen.

4.1.3 Nutzenbetrachtung

Der erwünschte bzw. angestrebte Nutzen der vorangegangenen Analyse einer Prozesslandschaft ist immer eine Verringerung von Kommunikationskosten. Diese soll vor allem bei unstrukturierten Prozessen nutzbringend eingesetzt werden, für die eine detaillierte Ablaufmodellierung nicht sinnvoll oder nicht möglich ist, für die aber trotzdem eine Analyse der Kommunikationskosten erwünscht ist.

Die Analyse der Infrastruktur einer Prozesslandschaft bezüglich ihrer Kommunikationskosten unterscheidet sich deutlich von in der Literatur beschriebenen (statischen) Analysen. Letztere untersuchen beispielsweise die „Senkung der Informations- und Kommunikationskosten für den Austausch von Informationen zwischen Akteuren ... durch die zunehmende Verbreitung und Nutzung von Standards“ [WSvW+00]. Andere untersuchen sogenannte kommunikationseffiziente parallele Algorithmen mit dem Ziel einer niedrigeren Kommunikationszeit als der lokalen Bearbeitungszeit. Hier sind jedoch die Anwendungsbereiche nicht Prozessmodelle, sondern Sortieralgorithmen und Multisearch-Verfahren [BD96, ABK95].

Ein besonderes Merkmal der im Rahmen des Process Landscaping durchgeführten statischen Analyse ist die Untersuchung einer sich aufgrund organisatorischer Maßnahmen verändernden Kommunikationsinfrastruktur. Diese organisatorischen Maßnahmen beziehen sich auf die Verlagerung von Aktivitäten, wie sie in der realen Welt durchaus regelmäßig vorkommen:

- Bei der Expansion eines Unternehmens am zentralen Standort kann es erforderlich werden, ein weiteres, zusätzliches Gebäude zu mieten.

- Nach einer Übergangszeit zieht ein Unternehmen oft von zwei kleineren in ein größeres Gebäude um.

- Es kommt vor, dass ein Standort verkleinert wird, weil beispielsweise der Mietvertrag eines Gebäudes nicht verlängert wird oder Aktivitäten ausgelagert wurden.

- Es sollte immer ein Bestreben sein, mehrfach vorhandene Aktivitäten gleichmäßig auszulasten. Dieses Bestreben kann beispielsweise über die Verlagerung von Verantwortlichkeiten realisiert werden und führt damit oft zu einem veränderten Kommunikationsverhalten.

Führt ein konkretes Optimierungsziel zu einer lokal begrenzten Kostensenkung innerhalb einer Prozesslandschaft, muss diese Kostensenkung unter Beachtung der möglichen Seiteneffekte auf andere Standorte der Prozesslandschaft betrachtet werden. In Abschnitt 4.1.2.2 sind entsprechende Situationen diskutiert worden. Dabei ist hervorzuheben, dass diese – einfach berechenbaren – Ergebnisse nicht durch bloßes Hinschauen zu erkennen sind, auch nicht durch genaues Betrachten der 269 in Anhang A.2 aufgelisteten Kommunikationskanäle.

Eine Umsetzung der Ergebnisse sollte auch nicht ausschließlich aufgrund der gewonnenen Zahlenwerte durchgeführt werden. Es ist zuvor immer ihre Auswirkung auf die zugrundeliegenden realen Softwareprozesse zu untersuchen. Die Diskussion in Abschnitt 4.1.2.2 hat gezeigt, dass teilweise der zweitgünstigste berechnete Wert die bessere Alternative für den Ablauf der Softwareprozesse ist. Somit ist der Kkf nicht als ausschließliches Entscheidungskriterium anzusehen, jedoch als ein kostenbezogenes Hilfsmittel zur Entscheidungsfindung.

Zusammenfassend kann festgehalten werden, dass die Analyse statischer Kommunikationseigenschaften einer verteilten Prozesslandschaft auf eine Weise durchgeführt wurde, wie sie bislang in der bekannten Literatur noch nicht diskutiert worden ist. Dabei sei noch einmal daran erinnert, dass sich die Analyse auf fachlich sinnvoll strukturierte Prozesse konzentriert, um nutzbringende Ergebnisse anbieten zu können. Je stärker die statische Kommunikationsinfrastruktur für ein spezifisches Optimierungsziel verändert wird, desto nutzbringender ist die Analyse. Die Grenzen dieser Analyse sind jedoch dann erreicht, wenn der Schwerpunkt auf einer veränderten Nutzungshäufigkeit bereits vorhandener Kommunikationskanäle liegt oder lediglich verwendete Kostenmodelle variiert werden. Für derartige Situationen sollte einer dynamischen Analyse der Vorzug gegeben werden, wie sie in Abschnitt 4.2.2.2 diskutiert wird.

4.2 Analyse dynamischer Kommunikationseigenschaften

Die Analyse dynamischer Kommunikationseigenschaften einer Prozesslandschaft PL_{com} basiert auf der diskreten ereignisorientierten Simulation von zeitbehafteten Petrinetzen. Hierbei entspricht jedes Schalten einer Aktivität der Behandlung eines Ereignisses. Ablaufinformationen, die bei der Analyse statischer Kommunikationseigenschaften nicht berücksichtigt werden, sind damit eine unbedingte Voraussetzung für die Durchführung dieser Analysemethode. Während der Simulation einer Prozesslandschaft gilt jedes Senden, Verarbeiten und Empfangen einer Nachricht als ein Ereignis, welches die weitere Dynamik der gesamten Landschaft beeinflusst.

Allgemein unterscheidet man zwischen kontinuierlicher und diskreter Simulation [LK91]. Kontinuierliche Simulation wird meist dadurch realisiert, dass in konstanten, relativ kurzen Zeitabständen der Zustand eines simulierten Systems aktualisiert wird. Diskrete Simulation basiert auf der Annahme, dass sich ein Systemzustand nur durch Ereignisse ändert, die in Bezug zur Simulationszeit zu bestimmten, diskreten Zeitpunkten auftreten. Dabei kann jede Änderung des Systemzustandes wiederum zu einem Ereignis bzw. einer Menge von Ereignissen führen, so dass sich die Dynamik des Systems als ein Fortschreiben von Ereignissen darstellt. Da die Analyse dynamischer Kommunikationseigenschaften auf dem Nachrichtenaustausch (als Ereignis) zwischen Aktivitäten basiert, bietet sich hier die diskrete ereignisorientierte Simulation an.

Die in diesem Abschnitt diskutierte Simulation setzt den Fokus auf die Betrachtung der externen Kommunikation, die, gemäß dem Prinzip der teuren externen und vergleichsweise preiswerten internen Kommunikation, ausschlaggebend für die Kommunikationskosten einer Prozesslandschaft ist. Entsprechend sind einige Änderungen an der Beispiellandschaft vorgenommen worden, die nachfolgend in Abschnitt 4.2.1 erläutert werden.

4.2.1 Struktur der Prozesslandschaft $\mathbf{PL_{com}}$

Grundlage für die Analyse dynamischer Kommunikationseigenschaften ist, wie im Fall der Analyse statischer Eigenschaften, die lokale Sicht auf die Beispielprozesslandschaft, jedoch ergänzt um Ablaufinformationen. Die geografische Verteilung der Aktivitäten ist in Abbildung 4.2 dargestellt. Sie entspricht fast vollständig der Verteilung, wie sie in Abschnitt 4.1 diskutiert wurde. Ausnahmen bilden lediglich der separate Standort für das Domain Engineering innerhalb des zentralen Standortes und das Fehlen des Standortes *Kundenlokation*. Die dort vorhandenen Aktivitäten sind den Standorten *Anwendungsentwicklung* (Aktivität *Anforderungsanalyse erstellen*) und *Managementlokation* (Aktivitäten *Kommunikation mit dem Kunden* und *Abnahme mit dem Kunden*) zugeordnet worden. Der Grund für diese Verschiebung ist das Analyseergebnis in Abschnitt 4.1.2.2, demzufolge die Verlagerung der Aktivität *Anforderungsanalyse erstellen* zum Application Engineering eine deutliche Reduzierung der Kommunikationskosten bewirkt. Da die beiden anderen Aktivitäten *Kommunikation mit dem*

Kunden und *Abnahme mit dem Kunden* für die Betrachtung der externen Kommunikation zwischen den übrigen Standorten der Prozesslandschaft vernachlässigbar sind, konnten diese ebenfalls ohne weiteres verschoben werden. Dies hat die Auflösung des Standortes *Kundenlokation* zur Folge.

Ansonsten findet sich auch innerhalb der Prozesslandschaft PL_{com} an jedem Standort der obersten Abstraktionsebene das Component Engineering, welches überall identisch aufgebaut ist. Bis auf einen Teilbereich des Qualitätsmanagements am Standort *Außenstelle-1* finden alle weiteren Aktivitäten der komponentenbasierten Softwareentwicklung am zentralen Standort statt.

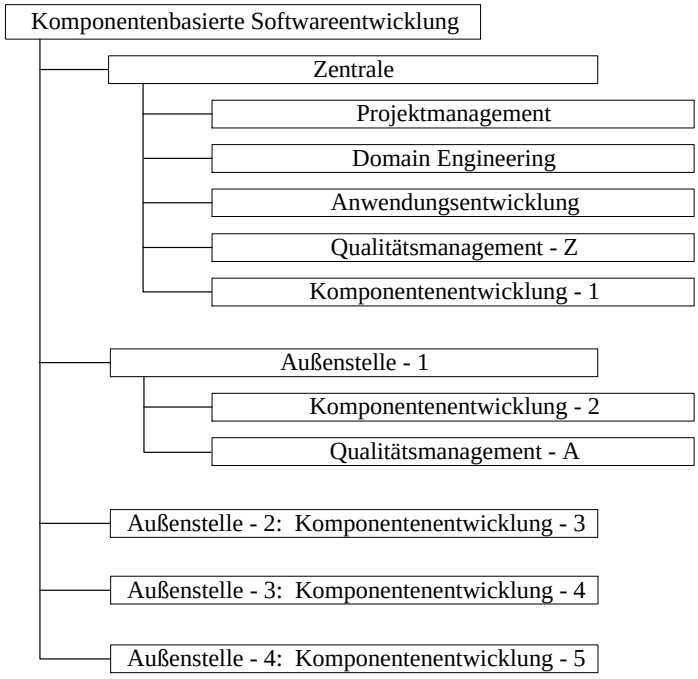


Abbildung 4.2: Lokale Sicht der Beispielprozesslandschaft PL_{com}

Für die Simulation der Beispielprozesslandschaft sind dieser einige Aktivitäten hinzugefügt worden, die in unmittelbarem Zusammenhang mit der Kommunikation zwischen verschiedenen Standorten stehen. Sie unterstützen die Analyse dadurch, dass sie als Messpunkte für das Ablesen von Simulationswerten genutzt werden. Beispiele sind die Aktivitäten *Review Anforderungen anstoßen* am Standort *Anwendungsentwicklung* oder *Projekt freigeben* am Standort *Managementlokation*. Des Weiteren sind einige Aktivitäten zusammengefasst worden, die für die Betrachtung der externen Kommunikation der verschiedenen Standorte nicht relevant sind. So wird beispielsweise das Konfigurationsmanagement des Application Engineering und des Component Engineering als eine nicht weiter verfeinerte Aktivität angesehen. Ebenso werden verschiedene Entwicklungsaktivitäten des Application Engineering und des Component En-

gineering zusammengefasst, die hauptsächlich interne Kommunikation erzeugen. Im Component Engineering sind aus diesem Grund zum Beispiel alle Entwurfsaktivitäten zur Aktivität *Spezifikation erstellen* zusammengefasst. So wird gleichzeitig ein homogenes Abstraktionsniveau für alle modellierten Standorte erreicht, das als Ergebnis der Modellierungsphase nicht für alle Bereiche der Prozesslandschaft gegeben war. Dieses ist jedoch eine wichtige Voraussetzung, um zu sinnvoll verwertbaren Ergebnissen zu gelangen. Für eine detaillierte Auflistung aller Ergänzungen und Abstraktionen der Beispielprozesslandschaft sei auf [Stö01b] verwiesen.

Im zugehörigen Simulationsmodell der Beispielprozesslandschaft wird jeder Standort der unteren Ebene der lokalen Sicht als separates Petrinetz dargestellt. Die Standorte *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* sind dabei durch mehrere Instanzen eines Netzes abgebildet. Kommunikationskanäle werden ebenfalls in separaten Petrinetzen dargestellt. Jedes dieser nachfolgend als Kommunikationssystem bezeichneten Netze repräsentiert eine Menge von Kommunikationskanälen, die eine gemeinsame technische Realisierung besitzen, also gleiche Kapazitäts- und Kostenwerte aufweisen. Insgesamt besteht das Simulationsmodell aus 27 separaten Petrinetzen, auf die weit über hundert Aktivitäten und etwa ebenso viele Schnittstellen verteilt sind. Eine genaue Zählung der Aktivitäten und Schnittstellen wie in Abschnitt 4.1.1 (vgl. Tabelle 4.1 auf Seite 159) macht hier wenig Sinn, da mehrere Abstraktionen und Verfeinerungen speziell für die Simulation durchgeführt worden sind. Die konkrete Verteilung der Aktivitäten auf verschiedene Netze sowie ihre Parametrisierung ist in Anhang A.3 aufgeführt. Dort ist pro Standort das zugrundeliegende Petrinetz abgebildet. Zusätzlich ist jeder Abbildung eine Tabelle beigefügt, die Auskunft über die Dauer einer jeden Aktivität und über das Volumen jeder dargestellten Informationseinheit gibt.

Eine Besonderheit im Simulationsmodell stellt das Netz *Setup* dar. Es dient ausschließlich der Simulationssteuerung, indem es mit konkreten Werten belegte Instanzen der Petrinetze für die einzelnen Standorte anlegt und über Referenzen bestimmt, welche Instanzen während der Simulation jeweils miteinander kommunizieren. Außerdem produziert es in zufällig festgelegten Zeitabständen jeweils eine Marke zum Erzeugen einer Kundenanfrage bzw. eines Kundenauftrags. Die Marken werden an das Netz des Projektmanagements übertragen und starten so ein (nächstes) Softwareentwicklungsprojekt.

Beim Erzeugen der Auftragsmarken wird ihr Komplexitätswert – und damit die Komplexität eines Softwareentwicklungsprojektes – gleichverteilt zwischen 0 und 10 zufällig gewählt. Die obere Grenze von 10 ist hier nicht zufällig entstanden, sondern hat sich im Verlauf vieler Gespräche mit verschiedenen Softwareprojektleitern und -mitarbeitern als praktisch erwiesen. Der Komplexitätswert, der sich im Verlauf der Simulation nicht mehr verändert, beeinflusst die Zeitbedarfe aller nachfolgenden Aktivitäten, das Volumen von Nachrichten und Bearbeitungsentscheidungen. Die Grundlagen dieser Werte beruhen auf der Annahme, dass mit der modellierten Prozesslandschaft verschiedene Softwareprojekte in einer Größenordnung von ca. 100, 500 und 1000 Personentagen simuliert werden.

Der Zeitbedarf der Aktivität *Anforderungsanalyse erstellen* wird beispielsweise mit Hilfe der Funktion $4 + x.c() \times 4,1$ berechnet. Dabei repräsentiert 4 die Mindestdauer dieser Aktivität, x repräsentiert den Komplexitätswert, und $x.c()$ ist eine Verzögerungsfunktion zur Bestimmung des Zeitbedarfes der Aktivität. Der Faktor 4,1 dient der Modellierung der Spanne zwischen minimalem und maximalem Zeitbedarf der Aktivität *Anforderungsanalyse erstellen*. So wird der Zeitbedarf einer Aktivität durch eine Verzögerungsfunktion, eine additive und eine multiplikative Konstante berechnet. In ähnlicher Form wird das Volumen von auszutauschenden Informationen – insbesondere von Softwareartefakten – berechnet, das ebenfalls von der Größe der jeweiligen Softwareprojekte abhängig ist.

Die konkreten Werte der so parametrisierten Prozesslandschaft sind in Anhang A.3 ab Seite 272 detailliert aufgeführt. Dort finden sich auch verschiedene Kostenmodelle, die zur Berechnung der Kommunikationskosten bei der Nutzung der Kommunikationskanäle benötigt werden. Die Parametrisierung der Prozesslandschaft ermöglicht die Analyse der in in Abschnitt 2.3.3 eingeführten Kommunikationseigenschaften. Für den Zusammenhang zwischen ihnen und den zu messenden Attributen sei noch einmal auf Abbildung 2.7 auf Seite 41 verwiesen.

Nach der Erläuterung der Struktur der Beispielprozesslandschaft PL_{com} und ihrer Abbildung im Simulationsmodell werden im nachfolgenden Abschnitt die Simulationsergebnisse dieser Ausgangsbasis diskutiert.

4.2.2 Analyse der Kommunikationseffizienz einer Prozesslandschaft $\mathbf{PL_{com}}$

4.2.2.1 Analyse der Ausgangsbasis

Die Kommunikationseffizienz einer Prozesslandschaft PL_{com} wird im Rahmen der Simulation an ihren zentralen Eigenschaften Kommunikationsaufkommen, Prozessauslastung, durchschnittliche Pfadlänge und Initiator-/Responder-Erfüllungsrate gemessen (vgl. Abschnitte 2.3.3 und 2.3.5). Für die Auswertung der konkreten Simulationsergebnisse ist zu beachten, dass pseudozufällige Ereignisse das Systemverhalten bestimmen [LK91]. Für Petrinetze bedeutet das, dass bei einem Simulationslauf nicht zwangsläufig jeder Pfad auf dem Erreichbarkeitsgraphen durchlaufen wird. Jeder Simulationslauf stellt damit eine Realisierung eines stochastischen Prozesses dar. Um für diesen aussagekräftige Analysen zu erstellen, sind mehrere Simulationsläufe unter gleichen Rahmenbedingungen erforderlich. Für die Analyse der Ausgangsbasis und auch der später betrachteten Optimierungsansätze wird daher eine Hierarchie von Projekten, Experimenten und Simulationsläufen aufgebaut, wie sie in Abbildung 4.3 skizziert ist.

Eine Menge von Simulationsläufen wird in einem Experiment zusammengefasst. Jedes Experiment beschreibt die vollständige Parametrisierung eines Simulationsmodells, in diesem Fall die Parametrisierung der Elemente aller 27 Petrinetze mit insgesamt 34 Simulationsläufen. Auf der Projektebene können wiederum mehrere Experimente vor-

handen sein, die verschiedene Ausprägungen einer Prozesslandschaft, beispielsweise für verschiedene Optimierungsansätze, beinhalten.

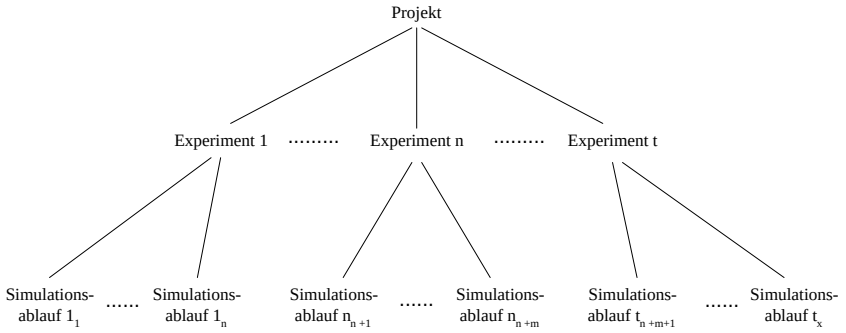


Abbildung 4.3: Hierarchie von Projekten, Experimenten und Simulationsläufen

Für die Auswertung aller anfallenden Daten innerhalb eines Experimentes ist eine Werkzeugunterstützung entwickelt worden, die

- Kommunikationskosten zuweist,
- Prozessauslastungen berechnet,
- Pfadlängen bestimmt und
- Initiator-/Responder-Raten ermittelt.

Auf die Werkzeugunterstützung wird in Kapitel 5 noch näher eingegangen. Dieser Abschnitt konzentriert sich auf die Diskussion der Simulationsergebnisse. Diese werden als Realisierungen einer (unbekannten) Verteilung aufgefasst. Um Letztere näher zu beschreiben, werden ein erwartungstreuer Mittelwertschätzer und eine Standardabweichung als stochastisch verteilte Größen berechnet [BS89]. Ihre Darstellung im laufenden Text erfolgt in folgender Schreibweise:

$$< erwartungstreuer Mittelwertschätzer > \pm < Standardabweichung >$$

Die in den nachfolgenden Tabellen 4.12 bis 4.18 aufgeführten Werte der berechneten Minima, Maxima, Mittelwerte und Standardabweichungen sind jeweils über alle 34 Simulationsläufe gemittelt.

Die nachfolgende Diskussion ist entlang der betrachteten Eigenschaften strukturiert. Für alle gilt jedoch, dass der Analyseschwerpunkt auf der Kopplung zwischen den einzelnen Bereichen der Zentrale und den Außenstellen der Prozesslandschaft liegt. Dabei sind in den Übersichtstabellen nicht immer alle berechneten Werte aufgeführt, sondern lediglich diejenigen, auf die in der Diskussion näher eingegangen wird.

Kommunikationskosten:
Bei der Betrachtung der Kommunikationskosten ist zu beachten, dass im Rahmen der dynamischen Analyse Kosten für externe Kommunikation gemäß dem Verursacherprinzip immer dem Sender angelastet werden. Die in Tabelle 4.12 aufgeführten Kosten berechnen sich als Summe der durch die Nutzung unterschiedlicher Kommunikationssysteme anfallenden Gebühren. Dabei wird berücksichtigt, dass diese für relativ kurze Entfernungen z.B. innerhalb der Zentrale niedriger liegen als bei den größeren Entfernungen zwischen der Zentrale und den Außenstellen. So sind beispielsweise dem Standort *Projektmanagement* Kosten in Höhe von 11.317,66 $\pm$ 2.658,77 Euro vor allem dadurch entstanden, dass er Entwicklungsaufträge an die weiter entfernten Standorte *Komponentenentwicklung-2* bis *Komponentenentwicklung-5* vergibt.

Ort	**Min**	**Max**	**erwartungstr. Mittelwert**	**Standard-abweichung**
Anwendungsentwicklung	309,10	8092,43	4999,36	1547,79
Komponentenentwicklung-2	66,69	653,13	252,85	122,37
Komponentenentwicklung-3	2250,60	4524,39	19988,44	11236,82
Komponentenentwicklung-4	2822,33	45039,23	19616,26	10263,10
Komponentenentwicklung-5	3126,34	38423,55	19446,60	9492,91
Projektmanagement	4283,68	17481,06	11317,66	2658,77
Qualitätsmanagement-Z	7341,68	51663,98	30186,45	8525,46
Qualitätsmanagement-A	0,00	1138,77	56,58	222,11

Tabelle 4.12: Kommunikationskosten der Beispielprozesslandschaft

Die Kommunikationskosten zwischen dem Component Engineering und dem Qualitätsmanagement fallen teilweise als interne Kosten an, beispielsweise zwischen *Komponentenentwicklung-1* und *Qualitätsmanagement-Z* in der Zentrale oder zwischen *Komponentenentwicklung-2* und *Qualitätsmanagement-A* in der Außenstelle-1. Für die *Komponentenentwicklung-3* bis *Komponentenentwicklung-5* sind diese Kosten als teure externe Kommunikationskosten zu berechnen. In Tabelle 4.12 können die beiden Situationen direkt miteinander verglichen werden. Dem zentralen Qualitätsmanagement werden Kosten in Höhe von 30.186,45 $\pm$ 8.525,46 Euro zugeordnet, den mit ihm kommunizierenden und weit entfernt liegenden Standorten *Komponentenentwicklung-3* bis *Komponentenentwicklung-5* Gesamtkosten in Höhe von ca. 59.000 Euro. Im Vergleich zum zentralen Qualitätsmanagement wird das Qualitätsmanagement der Außenstelle-1 lediglich mit 56,58 $\pm$ 222,11 Euro für seine externe Kommunikation belastet, wenn es Vorschläge zur Richtlinienerweiterung an die Zentrale schickt. Die hohe Streuungsrate liegt darin begründet, dass dieser Versand nur relativ selten erfolgt. An dieser Stelle sei darauf hingewiesen, dass die Standardabweichung gleich der Wurzel aus dem erwartungstreuen Varianzschätzer (Wurzel der Summe der Fehlerquadrate) [BS89] ist und damit auch bei einem gemittelten Simulationsergebnis, dessen Standardabweichung größer ist als der erwartungstreue Mittelwert, keine negativen Werte auftreten.

Insgesamt weist das zentrale Qualitätsmanagement die höchsten Kommunikationskosten innerhalb der Beispielprozesslandschaft PL_{com} auf. Ein Vergleich mit der Außenstelle-1 lässt jedoch vermuten, dass diese Kosten durch eine geeignete Umverteilung der Arbeitslast wahrscheinlich verringert werden können.

Prozessauslastung:

Für eine an einem (beliebigen aber festen) Standort stattfindende Aktivität wird die Prozessauslastung zu einem bestimmten Zeitpunkt t gemessen. Ein Standort gilt als ausgelastet, solange mindestens ein Nachbereich einer Aktivität mit einer Marke belegt ist, dessen Zeitstempel größer als t ist. Diese Auslastung wird im Verhältnis zur Gesamtzeit eines Softwareprojektes auf einer Skala von 0 bis 100 in Prozent gemessen.

Ort	**Min**	**Max**	**erwartungstr. Mittelwert**	**Standard-abweichung**
Anwendungsentwicklung	63,28	88,44	76,50	6,32
Komponentenentwicklung-1	16,18	66,37	46,03	13,78
Komponentenentwicklung-2	25,55	78,04	53,41	0,14
Komponentenentwicklung-3	32,13	74,50	50,94	9,04
Komponentenentwicklung-4	27,59	75,81	51,80	11,38
Komponentenentwicklung-5	27,11	79,86	53,97	12,42
Domain Engineering	28,33	51,00	37,73	5,28
Projektmanagement	20,31	54,12	35,38	7,39
Qualitätsmanagement-Z	66,22	98,65	85,84	9,09
Qualitätsmanagement-A	1,05	20,90	10,28	5,53

Tabelle 4.13: Prozessauslastung der Beispielprozesslandschaft

Tabelle 4.13 zeigt die berechneten Auslastungen für die verschiedenen Standorte der Beispielprozesslandschaft. Bei ihrer Bewertung ist zu berücksichtigen, dass die Auslastungen aller Standorte derart zusammenhängen, dass beispielsweise ein stark beanspruchter Standort die Auslastung nachfolgender Aktivitäten an anderen Standorten negativ beeinflussen kann. Das ist dann der Fall, wenn die nachfolgenden Aktivitäten auf Informationen vom ausgelasteten Standort warten. Daher sind die Werte in Tabelle 4.13 immer in ihrem Gesamtzusammenhang zu betrachten.

Der Standort *Anwendungsentwicklung* hat mit 76,50 $\pm$ 6,32% erwartungsgemäß einen relativ hohen Auslastungsgrad. Dies liegt nicht nur darin begründet, dass hier ein großer Teil der produktiven Arbeit zur Entwicklung des Gesamtsystems durchgeführt wird, sondern ist vor allem durch die lange Nachlaufzeit während der Simulation, also der Zeit zwischen dem Eingang des letzten Entwicklungsauftrags und dem Beenden der letzten Aktivität *Zur Abnahme freigeben*, zu erklären.

Die Standorte *Domain Engineering* und *Projektmanagement* sind mit 37,73 $\pm$ 5,28% und 35,38 $\pm$ 7,39% relativ gering ausgelastet. Diese Auslastung liegt ebenfalls im erwarteten Rahmen, da sie dem zeitlichen Anteil der Aktivitäten am gesamten Soft-

wareentwicklungsprozess entspricht. Alle fünf Standorte des Component Engineering weisen einen ungefähren Auslastungsgrad von etwas über 50± 12% auf. Dabei fällt vor allem die Höhe der Standardabweichung auf. Diese lässt sich durch die gleichverteilte Zuweisung der Projekte auf die einzelnen Standorte erklären, nach der jeder der fünf Standorte nur ein Fünftel der insgesamt durchzuführenden Aufträge erhält.

Das zentrale Qualitätsmanagement ist mit 85,84 ± 9,09% am stärksten ausgelastet, im Vergleich zu anderen Standorten sicher auch überlastet. Dies deutete sich bereits bei der Betrachtung der Kommunikationskosten an, die ebenfalls den höchsten Wert innerhalb der Beispielprozesslandschaft aufzeigen. Das Qualitätsmanagement der Außenstelle-1 weist dagegen mit 10,28 ± 5,53% einen sehr niedrigen Auslastungsgrad auf. Da es lediglich für einen Standort des Component Engineering zuständig ist, lässt dieser Wert auf eine daraus resultierende Unterbeschäftigung schließen. *Qualitätsmangement-Z* und *Qualitätsmanagement-A* zeigen somit eine sehr unterschiedliche und damit nicht optimale Auslastung der Beispielprozesslandschaft auf.

Durchschnittliche Pfadlänge:

In Abschnitt 2.3.5 ist die Kommunikationseffizienz bezüglich des Informationsflusses innerhalb verteilten einer Prozesslandschaft als möglichst große durchschnittliche Pfadlänge zwischen zwei verfeinerten Aktivitäten beschrieben, die zur Verarbeitung einer konkreten Information erforderlich ist. Da die Längen der während einer Simulation tatsächlich durchlaufenen Pfade zusammen mit der Häufigkeit ihres Auftretens in die in Tabelle 4.14 aufgeführten Werte eingehen, können diese analog zum Auslastungsgrad nur in ihrem Gesamtzusammenhang interpretiert werden. Die Bedeutung eines langen bzw. kurzen Pfades kann somit nur am konkreten, parametrisierten Simulationsmodell festgemacht werden. Aus Sicht eines Modellierers ist außerdem darauf zu achten, dass die Prozesslandschaft einen möglichst gleichverteilten Abstraktionsgrad aufweist, um die in einer Simulation gewonnenen Werte nicht zu verfälschen.

Ort	**Min**	**Max**	**erwartungstr. Mittelwert**	**Standard-abweichung**
Anwendungsentwicklung	2,2872	2,7774	2,6143	0,1401
Komponentenentwicklung-1	2,5455	3,3600	2,9480	0,2085
Komponentenentwicklung-2	2,8571	3,3333	3,1072	0,1390
Komponentenentwicklung-3	2,5000	3,3333	2,9891	0,1977
Komponentenentwicklung-4	2,4000	3,3600	3,0494	0,2084
Komponentenentwicklung-5	2,5882	3,3750	3,0652	0,1804
Domain Engineering	2,5455	3,3750	3,0320	0,2144
Projektmanagement	3,3759	3,5211	3,4443	0,0376
Qualitätsmanagement-Z	2,1691	2,9620	2,8596	0,1827
Qualitätsmanagement-A	2,0000	2,5833	2,3396	0,2118

Tabelle 4.14: Durchschnittliche Pfadlänge der Beispielprozesslandschaft

Eine statische Analyse der Pfadlängen lieferte einen Wertebereich von 1 bis 5; beide Extremwerte wurden im Projektmanagement identifiziert. Dieser Wertebereich lässt sich als Skala für die Interpretation der durch die Simulation gewonnenen Werte heranziehen. Eine durchschnittliche Pfadlänge im unteren Bereich der Skala weist damit auf eine geringe Bearbeitungstiefe hin, Werte im oberen Bereich auf eine vergleichsweise größere Bearbeitungstiefe der ausgetauschten Information. Für die genauere Betrachtung der Simulationsergebnisse sind nachfolgend jedoch nur diejenigen interessant, die auf eine geringe Bearbeitungstiefe hinweisen und nicht aufgrund einer tayloristischen Arbeitsorganisation entstanden sind (vgl. Abschnitt 2.3.5).

In Tabelle 4.14 weisen bis auf *Qualitätsmanagement-A* alle Standorte eine durchschnittliche Pfadlänge von über 2,5 und damit eine als nicht auffällig zu bewertende Bearbeitungstiefe auf. Für das Projektmanagement liegt dieser Wert mit $3{,}4443 \pm 0{,}0376$ am höchsten, obwohl hier in der statischen Analyse auch der kürzeste mögliche Pfad identifiziert wurde. Auf diesem werden Vorschläge zu Richtlinienerweiterungen an das zentrale Qualitätsmanagement weitergeleitet. Da dies aber nur relativ selten vorkommt, fällt die Pfadlänge von 1 hier kaum ins Gewicht.

Das Qualitätsmanagement der Außenstelle-1 weist mit einer durchschnittlichen Pfadlänge von $2{,}3396 \pm 0{,}2118$ als einziger Standort einen vergleichsweisen niedrigen Wert auf. Da hier jedoch bis auf das Erarbeiten von möglichen Richtlinienerweiterungen ausschließlich prüfende Tätigkeiten durchgeführt werden und daher eine tayloristische Arbeitsverteilung vorliegt, sind kurze Pfade durchaus gewollt. Eine niedrige durchschnittliche Pfadlänge ist also lediglich bei einer nicht tayloristischen Arbeitsverteilung ein Indiz für einen nicht optimalen Informationsfluss.

Initiator-/Responder-Erfüllungsrate:

Die Initiator-/Responder-Erfüllungsrate (kurz: I/R-Rate) dient als Indikator für die Kommunikationseffizienz bezüglich der Informationsverteilung innerhalb einer Prozesslandschaft. Diese Rate misst das anteilige Verhältnis erfolgreicher (ohne zeitliche Verzögerung durchgeführter) Kommunikationsversuche zur Gesamtzahl der während einer Simulation unternommenen Kommunikationsversuche. Die der Messung der I/R-Rate zugrundeliegende Modellierung bewirkt, dass ein Responder erst nach Durchlauf einer Reihe von Aktivitäten, die ein zu kommunizierendes Dokument als Ergebnis hat, wieder für eine Kommunikation verfügbar ist. Je mehr Zeit ein solcher Durchlauf in Anspruch nimmt, desto mehr Kommunikationsanfragen können nur zeitverzögert beantwortet werden. Damit hängt die I/R-Rate der verschiedenen Standorte eng mit ihrer jeweiligen Prozessauslastung zusammen.

Tabelle 4.15 zeigt die während der Simulation der Beispielprozesslandschaft gemessenen I/R-Raten derjenigen Standortpaare, für die der gemessene Wert eine nähere Betrachtung lohnt. Dies sind vor allem Werte in den Randbereichen des Intervalls von 0 bis 100 in Prozent. Für einige Paarungen wie beispielweise dem Application Engineering und einem Component Engineering ist die Betrachtung der I/R-Rate nicht relevant, da zwischen den beiden nur einmal pro Entwicklungsprojekt eine erstellte Komponente verschickt wird. Für andere Paarungen wie beispielsweise die Kommu-

Orte	Min	Max	erwartungstr. Mittelwert	Standard-abweichung
Anwendungsentwicklung ⇒ Qualitätsmanagement-Z	4,39	35,71	13,73	7,32
Qualitätsmanagement-Z ⇒ Anwendungsentwicklung	58,97	83,75	74,27	5,08
Domain Engineering ⇒ Anwendungsentwicklung	50,00	93,75	71,05	11,37
Domain Engineering ⇒ Qualitätsmanagement-Z	5,56	53,85	17,45	10,39
Projektmanagement ⇒ Qualitätsmanagement-Z	3,85	31,25	15,86	6,96
Qualitätsmanagement-Z ⇒ Projektmanagement	50,00	88,00	71,35	8,12
Projektmanagement ⇒ Domain Engineering	26,32	50,00	38,12	6,13
Domain Engineering ⇒ Projektmanagement	75,76	97,37	88,51	5,56

Tabelle 4.15: Initiator-/Responder-Erfüllungsrate der Beispielprozesslandschaft

nikation zwischen dem Projektmanagement und dem Domain Engineering mit einer I/R-Rate von 38,12 $\pm$ 6,13% (vgl. Tabelle 4.15) sind die Resultate relativ unspektakulär. Sie ermöglichen keine aussagekräftige Interpretation und werden daher nicht näher untersucht.

Wie aufgrund des hohen Auslastungsgrades des zentralen Qualitätsmanagements bereits zu erwarten war, weisen alle Paarungen, bei dem dieser Standort die Rolle des Responders einnimmt, mit einem Wert von maximal 17,45 $\pm$ 10,39% eine sehr geringe I/R-Rate auf: Nur maximal ein Viertel aller Kommunikationsversuche gelingt hier ohne zeitliche Verzögerung. Nimmt das zentrale Qualitätsmanagement dagegen die Rolle des Initiators ein, liegen die I/R-Raten bei über 70%. Alle übrigen Paarungen in Tabelle 4.15 haben – bis auf die Kommunikation vom Projektmanagement zum Domain Engineering – ebenfalls eine hohe Rate. Die Kommunikationseffizienz bezüglich des Informationsflusses ist damit als gut zu bewerten.

4.2.2.2 Analyse verschiedener Optimierungsansätze

Die in Abschnitt 4.2.2.1 diskutierte Auswertung der Beispielprozesslandschaft hat einige Schwächen und damit gleichzeitig einige Optimierungsansätze aufgezeigt. Letztere sind konform zu den allgemeinen Optimierungszielen, konkret

- einer Reduktion des externen Kommunikationsaufkommens,
- einer gleichmäßigeren Auslastung der Prozesse,

- einer Erhöhung der durchschnittlichen Bearbeitungstiefe und
- einer Steigerung der Rate erfolgreicher Kommunikationsversuche.

Für die identifizierten Optimierungsansätze wird nachfolgend ein Szenario entwickelt, für das durch die Simulation einer veränderten Parametrisierung der Beispielprozesslandschaft überprüft wird, ob dies im Vergleich zur ursprünglichen Prozesslandschaft tatsächlich zu einer verbesserten Effizienz der betrachteten Kommunikation bezüglich des *Kommunikationsaufkommens*, der *Prozessauslastung* und der *durchschnittliche Pfadlänge* führt. Dabei wird darauf geachtet, dass das Abstraktionsniveau der modifizierten Prozesslandschaft weiterhin einheitlich bleibt, um zu sinnvoll verwertbaren Ergebnissen zu gelangen. Die veränderte I/R-Rate wird für das modifizierte Simulationsmodell nicht näher betrachtet, da die Auswertung der Ausgangsbasis gezeigt hat, dass deren Analyse eng mit der der Prozessauslastung zusammenhängt. Die veränderten Werte sind jedoch der Vollständigkeit halber in Anhang A.3 aufgeführt (vgl. Tabelle A.79 in Anhang A.3).

Szenario:
Die Auswertung der Kommunikationskosten in Abschnitt 4.2.2.1 hat gezeigt, dass diese für die Kommunikation zwischen den Standorten *Qualitätsmanagement-A* und *Komponentenentwicklung-2* in der Außenstelle-1 deutlich unter denen für die Kommunikation zwischen dem zentralen Qualitätsmanagement und den an der Außenstelle-2 bis Außenstelle-4 vorhandenen Komponentenentwicklungsstandorten liegen. Um die Kosten des zentralen Qualitätsmanagements zu senken, soll jetzt über ein VPN (Virtual Private Network) [BGKK99] mit den weit entfernt liegenden Komponentenentwicklungsstandorten kommuniziert werden. Das VPN arbeitet auf Basis des Internets. Die Kommunikationspartner müssen sich für seine Nutzung lediglich zum Ortstarif bei ihrem lokalen Internetprovider einwählen. Auf diesem Weg sollten die Kommunikationskosten erheblich verringert werden können.

Auch die Prozessauslastung der Ausgangsbasis hat eine große Diskrepanz zwischen dem zentralen Qualitätsmanagement und dem der Außenstelle-1 aufgezeigt. Für eine gleichmäßigere Auslastung soll die Betreuung der Standorte *Komponentenentwicklung-3* bis *Komponentenentwicklung-5* vom Standort *Qualitätsmanagement-A* übernommen werden. Dieses Szenario entspricht damit dem der statischen Analyse in Abschnitt 4.1.2.2, bei dem ebenfalls die Auswirkung der Verantwortlichkeitsverlagerung für mehrfach in der Prozesslandschaft vorhandene Prüfaktivitäten untersucht wurde. Dort konnten jedoch allein auf Grundlage der Informationen über die statische Kommunikationsinfrastruktur und abstrakte Wertebereiche sowohl für die Kommunikationskosten als auch für die Nutzungshäufigkeiten der involvierten Kommunikationskanäle keine eindeutig belegbaren Verbesserungen identifiziert werden.

Für die Weiterleitung von Vorschlägen zur Richtlinienerweiterung vom Projektmanagement an das zentrale Qualitätsmanagement ist mit einer Pfadlänge von 1 der kürzeste Pfad innerhalb der gesamten Beispielprozesslandschaft identifiziert worden. Da diese Weiterleitung aber nicht dem Ansatz der tayloristischen Arbeitsteilung entspricht, soll-

te hier eine Möglichkeit zur Optimierung des Informationsflusses durch Umverteilung der Verantwortlichkeiten gegeben sein. Dazu sollen die Richtlinien im modifizierten Modell von beiden Qualitätsmanagementstandorten gleichberechtigt verändert werden dürfen. Aus diesem Grund werden die Richtlinien auf einen Server ausgelagert, auf den beide Standorte bzw. deren Qualitätsmanagementaktivitäten schreibenden Zugriff haben. Der Pfad zur Weiterleitung von Vorschlägen zur Richtlinienerweitung entfällt damit völlig. Gleichzeitig ermöglicht die Auslagerung einen lesenden Zugriff für alle Aktivitäten des Component Engineering, womit auch die – relativ häufig vorkommenden – Anfragen nach Richtlinien seitens der Standorte *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* und *Anwendungsentwicklung* entfallen.

Die Simulation des beschriebenen Szenarios und der Vergleich mit der ursprünglichen Prozesslandschaft führt zu folgenden Ergebnissen:

Kommunikationsaufkommen:
Das Kommunikationsverhalten der Komponentenentwicklungsstandorte an den Außenstellen-2 bis -4 hat sich zum einen bezüglich des Kommunikationspartners, zum anderen aber auch bezüglich des Kommunikationssystems und dem damit verbundenen Kostenmodell geändert. Eine Kommunikation zwischen ihnen und dem Qualitätsmanagement der Außenstelle-1 wird jetzt über eine ISDN-Verbindung mit dem Tarif RegioCall (vgl. Anhang A.3) aufgebaut. Das Resultat des so veränderten Simulationsmodells verursacht für den Standort *Qualitätsmanagement-A* erwartungsgemäß höhere Kosten, da dieser jetzt mehrere Standorte zu betreuen hat. Gleichzeitig sind jedoch die Kosten für das zentrale Qualitätsmanagement und auch für die betroffenen Komponentenentwicklungsstandorte erheblich gesunken. Insgesamt konnten die Kommunikationskosten für alle Qualitätsmanagement- und Komponentenentwicklungsstandorte von ursprünglich ca. 89.000 Euro um etwa 24.000 Euro und damit um mehr als ein Viertel gesenkt werden.

	Ausgangsbasis		**modifiziertes Modell**	
Ort	**erw. Mittelwert**	**Standard-abw.**	**erw. Mittelwert**	**Standard-abw.**
Anwendungsentwicklung	4999,36	1547,79	5167,68	1246,98
Komponentenentwicklung-2	252,85	122,37	243,18	95,55
Komponentenentwicklung-3	19988,44	11236,82	7773,99	4520,91
Komponentenentwicklung-4	19616,26	10263,10	7670,51	3647,21
Komponentenentwicklung-5	19446,60	9492,91	7316,71	3133,32
Projektmanagement	11317,66	2658,77	11204,86	2556,14
Qualitätsmanagement-Z	30186,45	8525,46	0,00	0,00
Qualitätsmanagement-A	56,58	222,11	952,42	256,62

Tabelle 4.16: Kommunikationskosten der modifizierten Beispielprozesslandschaft

Prozessauslastung:
Die gestiegenen Kommunikationskosten für den Standort *Qualitätsmanagement-A* sind auf die gleichzeitig gestiegene Prozessauslastung für diesen Bereich der Prozesslandschaft zurückzuführen. Der Auslastungsgrad liegt hier jetzt bei 64,10 $\pm$ 5,62% (vgl. Tabelle 4.17). Damit verbunden ist die deutliche Verbesserung der Auslastung für das zentrale Qualitätsmanagement. Sie ist von ursprünglich 85,84 $\pm$ 9,09% auf 69,22 $\pm$ 11,42% gesunken. Beide Standorte weisen nunmehr einen akzeptablen Auslastungsgrad auf.

	Ausgangsbasis		**modifiziertes Modell**	
Ort	**erw. Mittelwert**	**Standard-abw.**	**erw. Mittelwert**	**Standard-abw.**
Anwendungsentwicklung	76,50	6,32	77,71	7,12
Komponentenentwicklung-1	46,03	13,78	51,94	12,64
Komponentenentwicklung-2	53,41	0,14	46,25	12,03
Komponentenentwicklung-3	50,94	9,04	45,40	18,39
Komponentenentwicklung-4	51,80	11,38	46,54	15,98
Komponentenentwicklung-5	53,97	12,42	50,79	12,09
Domain Engineering	37,73	5,28	42,22	5,80
Projektmanagement	35,38	7,39	40,52	9,53
Qualitätsmanagement-Z	85,84	9,09	69,22	11,42
Qualitätsmanagement-A	10,82	5,53	64,10	5,62

Tabelle 4.17: Prozessauslastung der modifizierten Beispielprozesslandschaft

Insgesamt konnte – bis auf den Standort *Anwendungsentwicklung* und die Komponentenentwicklungsstandorte – eine Verbesserung für die gesamte Prozesslandschaft erzielt werden. Für die Komponentenentwicklungsstandorte ist dies nur natürlich, da ihr Kommunikationsverhalten nicht verändert wurde. Lediglich das Kostenmodell und die Kommunikationspartner sind variiert worden. Für den Standort *Anwendungsentwicklung* ist die Prozessauslastung mit nunmehr 77,71 $\pm$ 7,12% noch etwas angestiegen. Der Wert ist ein deutliches Indiz für weiteren Optimierungsbedarf.

Durchschnittliche Pfadlänge:
Die Betrachtung der durchschnittlichen Pfadlänge für die modifizierte Prozesslandschaft ist lediglich für das Projektmanagement und die Standorte *Qualitätsmanagement-Z* und *Qualitätsmanagement-A* interessant, da nur von ihnen Anfragen nach Richtlinien weitergeleitet bzw. bearbeitet werden. Diese Anfragen waren im ursprünglichen Modell ausschlaggebend für extrem kurze Pfadlängen. Beim Projektmanagement wurden zudem im ursprünglichen Modell die Extremwerte 1 und 5 als Ergebnis einer statischen Analyse gemessen.

Tabelle 4.18 zeigt jedoch für das Projektmanagement nur eine sehr geringfügige Erhöhung der durchschnittlichen Pfadlänge. Dies lässt sich auf das seltene Auftreten von

	Ausgangsbasis		modifiziertes Modell	
Ort	erw. Mittelwert	Standard-abw.	erw. Mittelwert	Standard-abw.
Anwendungsentwicklung	2,6143	0,1401	2,5905	0,1135
Komponentenentwicklung-1	2,9480	0,2085	3,00	0,20
Komponentenentwicklung-2	3,1072	0,1390	3,0038	0,2026
Komponentenentwicklung-3	2,9891	0,1977	3,0144	0,1970
Komponentenentwicklung-4	3,0494	0,2084	3,09	0,19
Komponentenentwicklung-5	3,0652	0,1804	3,03	0,18
Domain Engineering	3,0320	0,2144	3,00	0,00
Projektmanagement	3,4443	0,0376	3,4571	0,0384
Qualitätsmanagement-Z	2,8596	0,1827	2,3053	0,0933
Qualitätsmanagement-A	2,3396	0,2118	2,2951	1,1914

Tabelle 4.18: Durchschnittliche Pfadlänge der modifizierten Beispielprozesslandschaft

Richtlinienanfragen zurückführen. Der entsprechende Wert für das Qualitätsmanagement der Zentrale ist dagegen gesunken. Die Ursache hierfür liegt in der Nutzung des Richtlinienservers, der jetzt anstelle einer Aktivität des Qualitätsmanagements für diesbezügliche Anfragen zur Verfügung steht. Pfade zur Bearbeitung von Richtlinien sind somit insgesamt verkürzt worden. Ähnliches gilt für das Qualitätsmanagement der Außenstelle-1. Hier ist der Unterschied zum Wert der durchschnittlichen Pfadlänge der Ausgangsbasis nur deshalb geringfügig, weil der Standort im modifizierten Modell Richtlinienerweiterungen jetzt nicht mehr über das Projektmanagement weiterleitet, sondern direkt auf den Server stellt. Die Pfadlänge ändert sich dadurch also nicht. Sie ließe sich zwar modellierungsseitig leicht durch das Hinzufügen oder Verfeinern von Aktivitäten ändern, dies würde jedoch das geforderte einheitliche Abstraktionsniveau derart verändern, dass keine sinnvoll verwertbaren Ergebnisse mehr berechnet werden könnten.

Bislang ist für die modifizierte Prozesslandschaft eine Verbesserung für das Kommunikationsaufkommen, die Prozessauslastung und die durchschnittliche Pfadlänge in Form der Zahlenwerte der Simulationsergebnisse diskutiert worden. Analog zur statischen Analyse müssen diese zahlenmäßigen Verbesserungen ebenfalls bezüglich ihrer Auswirkungen auf den realen Ablauf eines Softwareentwicklungsprojektes bewertet werden. Es ist zu untersuchen, ob die optimierten Zahlenwerte tatsächlich zu einer Verbesserung der Prozessabläufe führen oder ob die Steigerung der Kommunikationseffizienz auch Nachteile mit sich bringt.

Die durch das Szenario auf Seite 184 dargestellte Modifizierung der Prozesslandschaft beinhaltet sowohl einige Veränderungen der Kommunikationsinfrastruktur als auch Umverteilungen von Verantwortlichkeiten und veränderte Zugriffsmöglichkeiten auf bestimmte Daten. Besteht die Veränderung der Infrastruktur aus dem Austausch eines vorhandenen Kommunikationssystems durch ein kostengünstigeres wie im Fall des VPN, über das das zentrale Qualitätsmanagement jetzt mit den Außenstellen In-

formationen austauscht, so hat dies sicherlich keine negativen Seiteneffekte auf den Arbeitsablauf. Muss die bestehende Infrastruktur erweitert werden, damit das Qualitätsmanagement der Außenstelle-1 die Test- und Reviewtätigkeiten jetzt auch für die anderen Außenstellen durchführen kann, so bedeutet dies zunächst einen zusätzlichen Kostenaufwand. Für die vorliegende Situation kann dieser Aufwand jedoch als gering angesehen werden, der durch die berechneten Einsparungen schnell wieder ausgeglichen ist.

Viel interessanter aber ist die Frage, welche Auswirkungen die Umverteilung der Verantwortlichkeiten vom zentralen Qualitätsmanagement auf die fachliche Strukturierung der Prozessabläufe der Außenstelle-1 hat. Ist mit der Außenstelle-1 an jedem Softwareprojekt beteiligt und ist damit auch ständig das dort angesiedelte Qualitätsmanagement-A involviert, so ist auch die Umverteilung der Verantwortlichkeiten sinnvoll. Ist jedoch die Außenstelle-1 nur an einigen wenigen Softwareprojekten beteiligt, so kann das zentrale Qualitätsmanagement nur für diese Projekte entlastet werden, und eine permanente Veränderung der Verantwortlichkeiten wäre die Folge. Eine solche Situation ist eher nicht sinnvoll, denn ein ständig wechselnder Prozessablauf trägt nicht zur Erhaltung oder gar Steigerung der Prozessqualität bei.

Schließlich bedeutet die Auslagerung der Richtlinien auf einen Server, auf den alle Beteiligten zumindest einen lesenden Zugriff haben, eine Veränderung von Zugriffsmöglichkeiten, die sich positiv auf den Ablauf einer Softwareentwicklung auswirkt. So entfallen viele Anfragen – und damit insbesondere auch der Aufwand für deren Beantwortung –, die sich negativ auf die Prozessauslastung ausgewirkt haben. Mit der modifizierten Situation ist den zuvor mit der Beantwortung beschäftigten Aktivitäten jetzt mehr Raum für die Ausführung ihrer eigentlichen Aufgaben gegeben.

Insgesamt sind damit die Ergebnisse der dynamischen Analyse ebenso wie die der statischen immer vor dem Hintergrund der modellierten realen Prozesse zu sehen und zu bewerten, damit sie zu einer echten Verbesserung führen.

4.2.3 Nutzenbetrachtung

Der angestrebte Nutzen bei der Analyse dynamischer Kommunikationseigenschaften auf der Basis einer Simulation ist die Erreichung einer möglichst hohen Kommunikationseffizienz der untersuchten verteilten Prozesslandschaft. Diese Effizienz bezieht sich auf

- den Auslastungsgrad verteilter Aktivitäten mit Blick auf die Kommunikationshäufigkeit,
- die räumliche Verteilung von Aktivitäten bezüglich entstehender Kommunikationskosten,
- die Informationsbereitstellung mit Auswirkung auf die Zugriffshäufigkeit auf Informationen, die von Aktivitäten explizit verteilt werden und

- die Minimierung der Anzahl vorhandener kurzer Informationsflüsse bzw. Bearbeitungspfade bezüglich Informationen gleichen Typs.

Der in Abschnitt 4.2.2.2 diskutierte Ansatz greift damit die Prozesssimulation und -optimierung als eines der von Fugetta angeführten wesentlichen Anwendungsgebiete der Prozessmodellierung auf [Fug00]. Zusätzlich zu den dort genannten Zielen des Erkennens unnötiger Verzögerungen und der Umverteilung von Verantwortlichkeiten berücksichtigt er auch Ziele einer verbesserten Kommunikationskostenstruktur und einer besseren Prozessauslastung. Bezogen auf eine Prozesslandschaft sind diese Ziele erreichbar durch die Identifikation

- derjenigen Stellen, an denen die Effizienz von Aktivitäten durch häufiges Warten auf eine Beantwortung ihrer Anfragen vermindert wurde. Im Beispiel wurde durch die Auslagerung der Qualitätsrichtlinien auf einen Server eine Verbesserung erreicht, da auf diesem Weg sowohl die Anzahl wartender Aktivitäten als auch die Dauer vieler Wartezeiten stark verringert werden konnte.
- derjenigen Stellen, an denen sich die Einführung eines preiswerteren Kommunikationssystems lohnt. Eine Kostenersparnis für die Beispielprozesslandschaft durch die Einführung eines neuen Kommunikationssystems konnte ebenfalls in Abschnitt 4.2.2.2 aufgezeigt werden.
- von bezüglich ihrer Auslastung über- und unterforderten Aktivitäten. Die Umverteilung von Aufgaben bzw. Verantwortlichkeiten bezüglich des Qualitätsmanagements führte zu einer insgesamt besseren Auslastung aller Standorte bzw. der an ihnen stattfindenden Aktivitäten.
- von Stellen geringer Bearbeitungstiefe, die sich zum Teil als erwünscht (tayloristische Arbeitsorganisation), zum Teil als verbesserungsbedürftig herausstellten. Für Letztere konnte in der Beispielprozesslandschaft eine Verbesserung vorgeschlagen und deren Nutzen (also eine tatsächliche Verbesserung) nachgewiesen werden.

Die Zielerreichung wurde in Abschnitt 4.2.2.2 an einer konkreten Beispielprozesslandschaft demonstriert. Zwar müssen auch hier – wie bei jeder Entscheidungsfindung – immer zusätzlich vorhandene Rahmenbedingungen berücksichtigt werden, die Betrachtung der Kommunikationseffizienz hat sich jedoch insbesondere für verteilte Prozesslandschaften als nützliches Kriterium bei der Suche nach Optimierungsmöglichkeiten erwiesen.

Die in Abschnitt 4.2.2.2 genannten Ziele sind auch auf andere Prozesslandschaften anwendbar, da sie losgelöst vom konkreten Kontext der komponentenbasierten Softwareentwicklung formuliert sind. Bei der Interpretation der Auswertungsergebnisse sollte jedoch nicht vergessen werden, dass menschliches Verhalten mit modelliert wurde – zum Beispiel durch die Wahrscheinlichkeit, dass ein erstelltes Dokument nachbearbeitet werden muss (vgl. beispielsweise Aktivität *Anforderungsanalyse nachbessern*

am Standort *Anwendungsentwicklung*, Anhang A.3). Dies mindert nach Fugetta die Aussagekraft quantitativer Ergebnisse eines analysierten Modells [Fug00]. Daher sollte der Erkenntnisgewinn der Kommunikationsanalyse eher qualitativ als quantitativ gesehen bzw. bewertet werden. Dieser qualitative Erkenntnisgewinn wird durch den Vergleich der Simulationswerte für eine gegebene Prozesslandschaft mit einer für verschiedene Optimierungsziele modifizierten Prozesslandschaft erreicht, wie er in Abschnitt 4.2.2.2 vorgenommen wurde.

Sowohl die Analysemethoden des Process Landscaping als auch die dazu erforderliche vorangehende Modellierung von Prozesslandschaften werden in weiten Teilen durch geeignete Werkzeuge unterstützt. Diese Werkzeuge werden im nachfolgenden Kapitel näher vorgestellt. Sie sind teilweise eigens zur Unterstützung der Methode erstellt worden, teilweise ist bereits existierende Software um Process Landscaping-spezifische Komponenten erweitert worden.

Kapitel 5

Werkzeugunterstützung

Eine globale Anforderung an alle Werkzeuge, die im Rahmen des Process Landscaping eingesetzt werden, ist die Berücksichtigung der Petrinetze als formale Basis der Methode. Dabei sollte vor allem diejenige Petrinetz-Variante unterstützt werden, die von Ablaufinformationen und damit vom Kontrollfluss abstrahiert. Bislang konzentrieren sich aber sowohl Methoden der Prozessmodellierung als auch unterstützende Werkzeuge auf die Darstellung von Abläufen. Dies wird auf dem Gebiet der Softwaretechnologie bereits an der Definition des Prozessbegriffs deutlich, die einen Prozess als eine partiell geordnete Menge von Prozessschritten zur Erstellung eines Softwareproduktes bezeichnet und damit eine Vorgehensweise – also einen Ablauf – zur Durchführung konkreter Aktivitäten erkennen lässt (vgl. Kapitel 1.3). Viele Prozesse sind jedoch zu unstrukturiert oder folgen einer flexiblen Ablaufreihenfolge. Letztere beschränkt sich nicht auf Parallelitäten und Sequenzen von Aktivitäten, sondern lässt die Reihenfolge ihrer Durchführung offen. Diese Art von Prozessen kann daher bislang kaum oder gar nicht durch Prozessmodellierung unterstützt werden, denn jeder durch einen definierten Ablauf beschriebene Prozess unterstützt einen Prozessteilnehmer nicht nur, sondern schränkt gleichzeitig seine individuelle Vorgehensweise ein. Unstrukturierte oder flexibel ablaufende Prozesse – in der Literatur auch als semi-strukturiert bezeichnet [Dei00] – finden sich jedoch sehr häufig in Klein- und Mittelbetrieben, die verstärkt auf Kreativität und Flexibilität ihrer Softwareentwicklung setzen. Flexibilität bezieht sich hier auf eine nicht im vorhinein festgelegte Ablaufreihenfolge definierter Aktivitäten, die sich aus der Unstrukturiertheit des zugrundeliegenden Prozesses ergibt. Eine korrespondierende Flexibilität im Prozessmodell kann entweder durch die Modellierung aller möglichen Ablaufalternativen oder durch die Abstraktion von einem festen Ablauf ermöglicht werden. Für die Methode des Process Landscaping wird in Abschnitt 5.1 eine geeignete Werkzeugunterstützung für die zweite Möglichkeit vorgestellt.

Durch den aktuellen Trend, die Softwareentwicklung auf teilweise weit auseinanderliegende Standorte zu verteilen, entstehen weitere Anforderungen an eine werkzeuggestützte Prozessmodellierung: Er erfordert die Erstellung der Modelle durch mehrere Modellierer. Diese Erstellung muss trotz einer geografischen Verteilung sowohl

der Prozesse als auch der Modellierer in einem kooperativen Arbeitsprozess erfolgen [Zie96]. Mit solchen Kooperationsprozessen und der Notwendigkeit, diese informationstechnisch zu unterstützen, beschäftigt sich seit Mitte der achtziger Jahre das Forschungsgebiet der Computer-Supported Cooperative Work (CSCW) [Gre88]. Das in Abschnitt 5.1 diskutierte Werkzeug zur Unterstützung des Process Landscaping verfolgt mit der expliziten Berücksichtigung dieser arbeitsteiligen Kooperationsprozesse gleichzeitig eines der Kernziele des CSCW.

Durch ein verteiltes Modellieren von Prozessen ergibt sich des Weiteren die verstärkte Notwendigkeit einer werkzeugunterstützten Konsistenzprüfung für die erstellten Modelle. Verfügbare Modellierungswerkzeuge unterstützen ein verteiltes und kooperierendes Arbeiten bislang nicht oder nur unzureichend (vgl. beispielsweise Aris [IDS01], LeuSmart [ade99], Innovator [MID01], Bonapart [Int01], Adonis [BOC02]). Ein Team von Modellierern, das an einer gemeinsamen Menge von Prozessmodellen arbeitet, muss daher viel Aufwand für die Abstimmung der Schnittstellen zwischen den Einzelmodellen betreiben. Ein Werkzeug zur Prozessmodellierung sollte jedoch nicht nur die gemeinsame Erstellung einer Prozesslandschaft durch ein Modellierungsteam unterstützen, welches vielleicht zudem noch geografisch verteilt arbeitet, sondern auch eine gleichzeitige Nutzung der Prozessmodelle seitens der geografisch verteilten Prozessbeteiligten ermöglichen, die diese zur Unterstützung ihrer Arbeit verwenden wollen. Mit dieser Unterstützung ist hier nicht der Einsatz eines (verteilten) Systems gemeint, sondern eine Zugriffsmöglichkeit auf die Darstellung und Dokumentation der Prozesse für ein besseres Prozessverständnis.

Bislang konzentriert sich die Werkzeugunterstützung auf die Rolle des Modellierers, indem Funktionen zur einfachen und schnellen Darstellung von Prozessen angeboten werden. Prozessbeteiligte haben jedoch andere Anforderungen an die Nutzung einer Prozesslandschaft als ein Modellierer für deren Erstellung. Für sie ist es neben einem Gesamtüberblick wichtig, ohne große Umwege diejenigen Informationen innerhalb des Gesamtprozesses zu finden, die für sie relevant sind. Das bedeutet, dass die für ihren Standort und ihre Teilaufgaben relevanten Informationen einfach zu filtern sein müssen, Querbezüge zu anderen Aktivitäten aber über die modellierten Schnittstellen schnell herstellbar sein sollten.

Im Zusammenhang mit der Entwicklung des Process Landscaping ist ein Prozessmodellierungswerkzeug mit Namen *Piazza Palermo* entstanden [Pro01], mit dem eine zusammengehörige Menge von Prozessen verteilt und unter Vernachlässigung von Ablaufinformationen als eine einheitliche Prozesslandschaft kooperativ modelliert werden kann. Einige prototypisch realisierte Analysefunktionen stehen ebenfalls zur Verfügung. Prozessbeteiligte können über einen Webbrowser lesend auf eine modellierte Prozesslandschaft zugreifen. Ihnen stehen sowohl Navigationsmöglichkeiten durch die Landschaft als auch Möglichkeiten zur gezielten Informationssuche zur Verfügung. Die Oberfläche und die Funktionalitäten von *Piazza Palermo* werden in Abschnitt 5.1 vorgestellt.

Eine durchgängige weiterführende Modellierung der unteren Ebenen einer Prozesslandschaft ohne Bruch in der Anwendung des Process Landscaping wird mit Hilfe ei-

ner ebenfalls im Rahmen der Methode entwickelten Schnittstelle von *Piazza Palermo* zu einem bereits existierenden Prozessmodellierungswerkzeug gewährleistet [Bro02]. Diese Schnittstelle unterstützt den Modellierer Schritt für Schritt bei der konsistenten Erweiterung einer mit *Piazza Palermo* erstellten Prozesslandschaft zur Modellierung und Analyse ihrer unteren Ebenen. Sie wird in Abschnitt 5.2 ausführlich diskutiert.

Abschnitt 5.3 erläutert zunächst kurz die Vorgehensweise bei der Modellierung der unteren Ebenen einer Prozesslandschaft mit dem Prozessmodellierungswerkzeug *Renew* [KW99]. Der Fokus liegt dabei auf der Modellierung eines Simulationsmodells zur Analyse der dynamischen Kommunikationseigenschaften einer gegebenen Prozesslandschaft. Zur Auswertung der Simulationsergebnisse ist das Werkzeug im Rahmen dieser Arbeit um die Anbindung an eine Datenbank und eine entsprechende Auswertungskomponente erweitert worden, die speziell die Analyseziele des Process Landscaping berücksichtigt [Stö01b, SW02b]. Die Diskussion dieser Erweiterungen bildet einen weiteren Schwerpunkt des Abschnitts 5.3.

Alle Komponenten der Werkzeugunterstützung werden entlang der bereits aus den vorangegangenen Kapiteln bekannten Prozesslandschaft zur komponentenbasierten Softwareentwicklung diskutiert.

5.1 Modellierung und Analyse der oberen Ebenen einer Prozesslandschaft

Erklärtes Ziel bei der Realisierung des Prozessmodellierungswerkzeugs *Piazza Palermo* war die Erstellung eines Werkzeugs, welches insbesondere die rollenspezifischen Anforderungen von Modellierern und Prozessbeteiligten an eine Prozesslandschaft berücksichtigt. Der Zugriff auf die von *Piazza Palermo* zur Verfügung gestellten Funktionalitäten ist daher bezüglich der unterschiedlichen Rollenanforderungen aufgeteilt: Der Editor des Werkzeugs ist ausschließlich für eine Nutzung durch den bzw. die Prozessmodellierer vorgesehen. Er ermöglicht neben der grafischen Modellierung einer Prozesslandschaft auch deren Dokumentation und bietet Funktionen zur Konsistenzprüfung und Analyse. Die Trennung zwischen Funktionalitäten für Modellierung und Nutzung einer Prozesslandschaft seitens der Beteiligten wird durch einen zweiten, ausschließlich lesenden Zugriff auf die Daten einer Prozesslandschaft deutlich: Ein Viewer erlaubt jedem Prozessbeteiligten über einen Webbrowser lesenden Zugriff auf die Prozessdokumentation. Er realisiert gleichzeitig eine sehr einfache und effiziente Bereitstellung wichtiger Prozessinformationen für die geografisch verteilt arbeitenden Prozessbeteiligten.

Mit dieser Aufteilung der Kernfunktionen können sowohl Modellierer als auch Beteiligte der Prozesslandschaft individuell unterstützt werden. Beide Rollen greifen auf eine gemeinsame Datenbasis zu, die die Konsistenz der produzierten und genutzten Daten sichert. Bezüglich der zugrundeliegenden Softwarearchitektur sei an dieser Stelle lediglich darauf hingewiesen, dass *Piazza Palermo* client/server-basiert mit einer zentralen Datenbank arbeitet. Dies ist Grundlage für ein verteiltes Arbeiten sowohl der

Modellierer als auch der Prozessbeteiligten. Die Architektur wurde an die Struktur des Model-View-Controller-Entwurfsmusters [GHJV94] angelehnt. Eine detaillierte Beschreibung zur Verteilung der einzelnen Softwarekomponenten auf Client und Server und ihrer Kommunikation untereinander ist in [Pro01] zu finden.

5.1.1 Unterstützung der Modellierungsphase

Im Folgenden wird erläutert, wie die Aufgaben und Anforderungen eines Prozessmodellierers unterstützt werden bzw. durch entsprechende Funktionen im Editor des Werkzeugs realisiert worden sind.

Abbildung 5.1 zeigt eine Prozesslandschaft, in der Kernprozesse der komponentenbasierten Softwareerstellung abgebildet sind. Alle zu einer Prozesslandschaft gehörigen Prozesse, Teilprozesse und Aktivitäten sind hierarchisch geordnet als Baum dargestellt. Ihre Relation zueinander – konkret ihre hierarchische Ordnung – ist aus dem Navigationsbaum, dem sogenannten Aktivitätenbaum, ersichtlich. Von den auszutauschenden Informationen ist in dieser Darstellung abstrahiert worden.

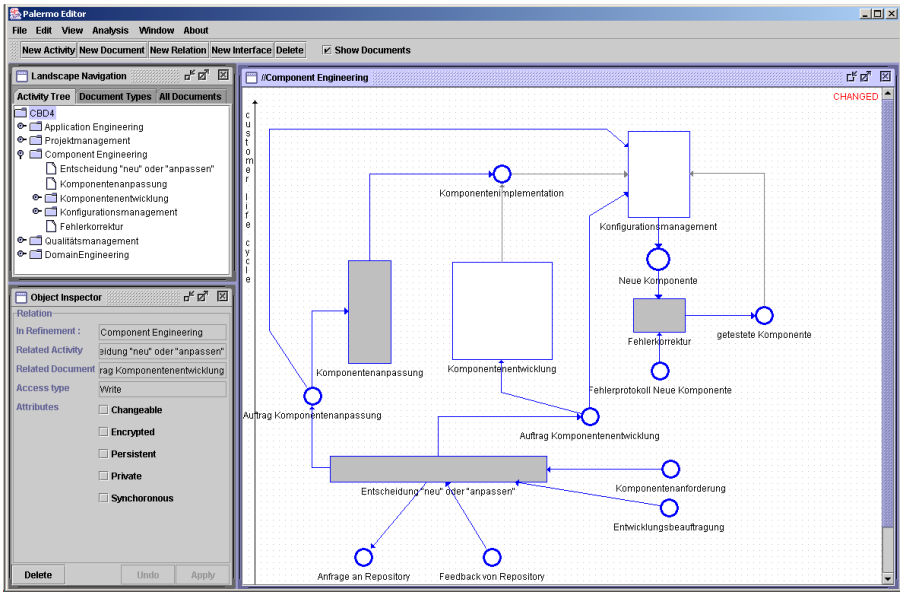


Abbildung 5.1: Verschiedene Arbeitsfenster zur Darstellung und Bearbeitung einer Prozesslandschaft

Zusätzlich bietet das Werkzeug dem Modellierer die Möglichkeit, alle innerhalb der Prozesslandschaft definierten Dokumente (Registerkarte *All Documents*) und deren Typen (Registerkarte *Document Types*) angezeigt zu bekommen. Als Beispiel sei das in der hier diskutierten Prozesslandschaft dargestellte Dokument *Entwicklungsbeauftragung* vom Typ *Auftragsdokument* genannt. Bei der Modellierung einer Prozesslandschaft mit *Piazza Palermo* wird davon ausgegangen, dass alle Informationen über Do-

kumente ausgetauscht werden, die jeweils von einem konkreten Typ wie z.B. *Anforderungsdokument* oder *Review-Bericht* sind. Es wird also nicht nur von Ablaufinformationen innerhalb der Prozesse abstrahiert, man beschränkt sich auch bei der Modellierung von Informationsobjekten auf diejenigen Kerndaten, die schriftlich fixiert werden und einer konkreten Dokumentationsform (Dokumenttyp) zugeordnet werden können.

Abbildung 5.1 zeigt alle zur Aktivität *Component Engineering* gehörenden Teilaktivitäten (dargestellt durch Rechtecke) und verwendete Dokumente (dargestellt durch Kreise). Der Überblick über die gesamte Softwareprozesslandschaft bleibt durch das Fenster *Landscape Navigation* erhalten. Im rechts dargestellten Arbeitsfenster können die Landschaftselemente editiert werden. Jede Aktivität kann zusätzlich noch weiter verfeinert werden. Bereits verfeinerte Aktivitäten innerhalb einer Abstraktionsebene erkennt man im Arbeitsfenster an ihrer Weißfärbung. Nicht weiter verfeinerte Aktivitäten sind grau gezeichnet. In Abbildung 5.1 erkennt man sowohl verfeinerte als auch nicht verfeinerte Aktivitäten als Bestandteile der Aktivität *Component Engineering*.

Werden Dokumente von einer Aktivität geschrieben und von einer anderen Aktivität gelesen, bezeichnet man sie auch als Schnittstellendokumente. Liegen die beiden beteiligten Aktivitäten nicht in derselben Verfeinerung, werden die zugehörigen Schnittstellendokumente durch einen breiteren Rand besonders gekennzeichnet. In Abbildung 5.1 sind alle dort vorhandenen Dokumente Schnittstellendokumente, die sowohl von Aktivitäten innerhalb der Verfeinerung des Component Engineering als auch von Aktivitäten auf höheren oder niedrigeren Abstraktionsebenen verwendet werden.

Die grafische Repräsentation der Beispiellandschaft in Abbildung 5.1 bzw. der Teilausschnitt im rechten Arbeitsfenster erinnert an ein konventionelles Petrinetz, dargestellt durch einen bipartiten Graphen. Der Unterschied liegt im Fehlen des Kontrollflusses. Mit *Piazza Palermo* werden lediglich Zugriffe von Aktivitäten auf Dokumente (als spezielle Informationstypen) modelliert. Welche Aktivität jedoch wann und unter welchen Voraussetzungen auf Dokumente zugreift, ist nicht modelliert, da Ablaufinformationen nicht betrachtet werden. Eine mit *Piazza Palermo* erstellte Prozesslandschaft ist somit konform zu den in Abschnitt 3.1 formulierten formalen Grundlagen der oberen Ebenen einer Prozesslandschaft.

Durch die Abstraktion von Ablaufinformationen und die Fokussierung auf Aktivitäten, Informationen und den Zugriffen auf letztere ist das Modellierungsresultat bei der Verwendung von *Piazza Palermo* auch mit Datenflussdiagrammen vergleichbar, die laut Definition Verarbeitungen (hier: Aktivitäten) und Daten (hier: Dokumente) sowie die Verbindungen (hier: Lese-/Schreibzugriffe) zwischen beiden darstellen [Bal96].

Sind Veränderungen am Inhalt einer Verfeinerung vorgenommen worden, wird dies im zugehörigen Arbeitsfenster rechts oben durch den Begriff *CHANGED* angezeigt (vgl. Abbildung 5.1). Sämtliche Änderungen werden vorerst nur lokal am Arbeitsplatz des Modellierers durchgeführt und noch nicht in die Datenbank übernommen. Während der Modellierer seine Änderungen bzw. Erweitungen innerhalb einer Abstraktionsebene durchführt, wird dies allen anderen parallel an dieser Prozesslandschaft arbeitenden Modellierern angezeigt. Der Schreibschutz pro Arbeitsfenster und damit pro Abstraktionsebene einer Prozesslandschaft sichert die Konsistenz bei gleichzeitiger

Bearbeitung durch mehrere Modellierer. Bei einem Änderungsversuch eines zweiten Modellierers erhält dieser eine Fehlermeldung, die auch den Namen des bearbeitenden Modellierers enthält. Diese Information verhindert lange Wartezeiten bei notwendigen Änderungen und verbessert die Effizienz einer kooperierenden Modellierung, da die beteiligten Modellierer sich gegenseitig identifizieren und direkt (außerhalb des Werkzeugs) erforderliche Anpassungen an der Prozesslandschaft miteinander abstimmen können. Trotzdem bleibt im Unterschied zu anderen Prozessmodellierungswerkzeugen die parallele Bearbeitung einer Prozesslandschaft durch ein Modellierungsteam gewährleistet, da immer nur die jeweils bearbeitete Abstraktionsebene, d.h. nur die Verfeinerung einer einzelnen Aktivität, für den Schreibzugriff, nicht aber den Lesezugriff anderer Modellierer gesperrt ist.

Der Editor von *Piazza Palermo* erlaubt dem Modellierer jederzeit eine detaillierte Analyse der aktuellen Prozesslandschaft. So lassen sich früh Fehler während der Modellierung identifizieren oder Verbesserungen protokollieren. Der Aufruf der Analysefunktion liefert eine Menge verschiedener Ergebnisse, die analog zu den Aktivitäten einer Prozesslandschaft über einen Navigationsbaum angewählt werden können. Die oberste Ebene des Analysebaumes (vgl. *Statistics* in Abbildung 5.2) liefert zunächst allgemeine Informationen über eine Prozesslandschaft, wie beispielsweise die Anzahl modellierter Aktivitäten und Dokumente oder die maximale Tiefe des Aktivitätenbaumes.

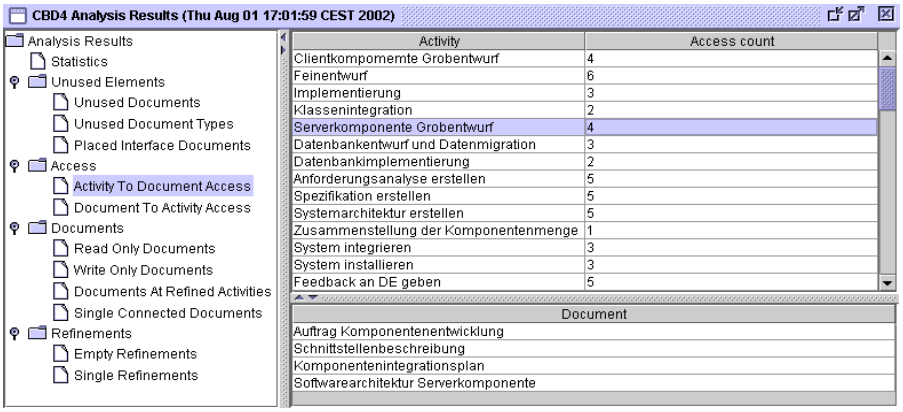


Abbildung 5.2: Analyseergebnis der Softwareprozesslandschaft

Abbildung 5.2 zeigt das Analyseergebnis der Beispiellandschaft. Die Kategorie *Unused Documents* liefert Informationen über diejenigen Landschaftselemente, die noch nicht über Zugriffsrelationen mit anderen verbunden sind. Die in der Abbildung ausgewählte Kategorie *Access* zeigt für jede Aktivität die Anzahl verwendeter Dokumente an. Durch Markieren einer konkreten Aktivität werden diese Dokumente jeweils namentlich aufgelistet. Diese Information kann auch über eine Dokumentensicht (*Document To Activity Access*) eingesehen werden. Über *Documents* kann man unter anderem erkennen, ob es modellierte Dokumente gibt, die nur gelesen oder nur geschrieben

werden. So sollten Dokumente wie z.B. das Anforderungsdokument im Laufe eines Softwareentwicklungsprojektes immer geschrieben und auch gelesen werden. Dagegen sollten etwa Programmierrichtlinien von Entwicklungsaktivitäten ausschließlich gelesen werden können. Ist aber aktuell nur ein schreibender Zugriff modelliert, kann dieser Fehler innerhalb der Statistik über *Write Only Documents* schnell identifiziert werden. Sollten Verfeinerungen einer Aktivität leer sein, oder enthält eine Verfeinerung nur eine einzige Aktivität, wird dies unter *Refinements* angezeigt.

Die Analyse unterstützt damit insbesondere Anforderungen, die der Rolle des Modellierers zuzuordnen sind. Dieser kann über die verschiedenen Analysekategorien auf schnellem und einfachem Weg Inkonsistenzen und Unvollständigkeiten in einer Prozesslandschaft identifizieren, die gerade bei der parallelen Modellierung eines verteilt arbeitenden Teams verstärkt auftreten können.

Eine Unterstützung semantischer Analysen wie z.B. die Untersuchung effizienter Verteilung der Prozesse und damit einer effizienten Kommunikationsstruktur für eine Prozesslandschaft $PL_{top-com}$ (vgl. Abschnitt 3.1.3) ist ein zukünftiges Ziel der Analyse mit *Piazza Palermo*. Dazu ist es bereits jetzt möglich, die modellierten Zugriffe von Aktivitäten auf Dokumente mit Kommunikationsattributen bezüglich Veränderbarkeit, Verschlüsselung, Persistenz, Privatheit und Synchronität zu versehen (vgl. *Object Inspector* in Abbildung 5.1). Die Auswertung der Attributwerte erfolgt momentan prototypisch über SQL-Abfragen; eine Umstrukturierung in die lokale Sicht muss noch manuell durchgeführt werden.

5.1.2 Unterstützung der Nutzungsphase

Dieses Unterkapitel beschreibt den Einsatz und Nutzen von *Piazza Palermo* für die Rolle des Prozessbeteiligten, der sich einen schnellen Überblick über den Gesamtprozess, der von ihm durchzuführenden Aktivitäten und die von ihm zu verwendenden Dokumente verschaffen will. Wie bereits erwähnt, ist für Prozessbeteiligte ein gesonderter, ausschließlich lesender Zugriff auf modellierte Prozesslandschaften implementiert worden. Dies hat den Vorteil, dass Prozessbeteiligte nicht von Modellierungsfunktionen des Werkzeugs abgelenkt werden oder sich aufwändig einarbeiten müssen. Zusätzlich schützt es die Arbeit der Modellierer vor ungewollten Veränderungen an einer Prozesslandschaft.

Piazza Palermo bietet über ein Webportal Zugriff auf den Aktivitätenbaum und die Dokumentation einer Prozesslandschaft. Dies erfordert lediglich einen Arbeitsplatzrechner, der über einen Internet-/Intranetanschluss und einen Browser verfügt. Der Editor von *Piazza Palermo* muss nicht verfügbar sein. Der Begriff des Webportals meint hier eine „Eingangspforte“ bzw. einen „Informationskanal“ für Prozessbeteiligte. Er erfüllt nicht unbedingt alle mit ihm verknüpften Anforderungen, die u.a. in [HH99] und [HKM01] diskutiert werden.

Abbildung 5.3 zeigt die diskutierte Softwareprozesslandschaft aus der Sicht eines Prozessbeteiligten, zu erkennen an ihrem Namen *CBD4* (Component-Based Development4), rechts oben in der Grafik aufgeführt. Der Aufbau des links in der Abbildung

dargestellten Navigationsbaumes ist an den des Editors angelehnt. Er erlaubt dem Anwender eine intuitive Navigation sowohl durch die Aktivitäten als auch durch die Liste modellierter Dokumente der aktuellen Prozesslandschaft. Wird beispielsweise eine Aktivität ausgewählt, so werden neben ihrer Dokumentation (*Description*) auch Querbezüge zu benötigten bzw. erzeugten Dokumenten (*Source/Target Documents*) aufgezeigt. Die Liste der zugehörigen Dokumente besteht aus Links, mit deren Hilfe direkt zu deren Beschreibung navigiert werden kann. Am Beispiel der Aktivität *Spezifikation erstellen* ist dies in Abbildung 5.3 gezeigt. Ist ein Anwender an der vollständigen Dokumentation einer Prozesslandschaft interessiert, kann er diese ebenfalls abrufen. Sie setzt sich zusammen aus den Einzeldokumentationen jeder Aktivität und jedes modellierten Dokuments.

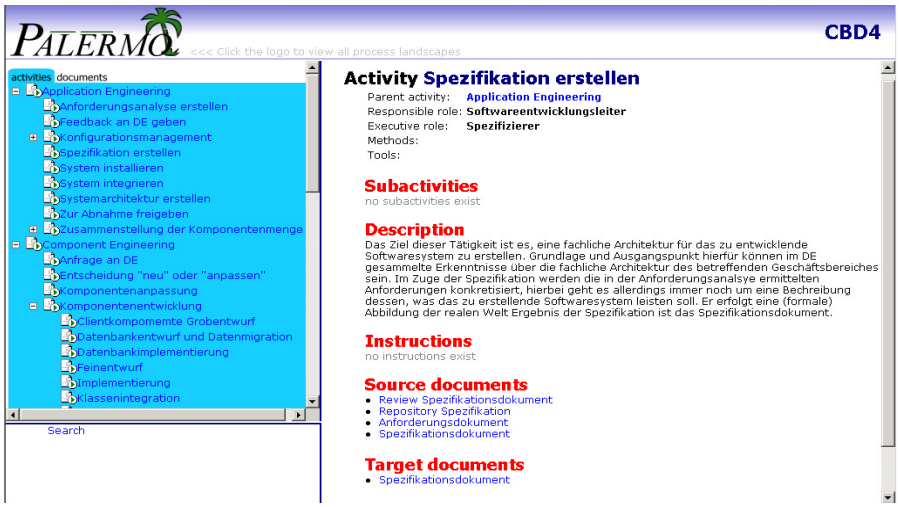


Abbildung 5.3: Webportal von *Piazza Palermo*

Das Portal von *Piazza Palermo* bietet neben dem manuellen Durchlauf durch eine Prozesslandschaft entlang des Aktivitätenbaumes und der Dokumentenliste auch jederzeit die gezielte Suche nach Aktivitäten oder Dokumenten. So erlaubt es allen Prozessbeteiligten eine effiziente und bequeme Nutzung ihrer Prozesslandschaft ohne die Notwendigkeit zusätzlicher Softwareinstallation. Aufwändige Einarbeitungsphasen oder gar Schulungen sind nicht notwendig, wenn dem Anwender der Umgang mit dem Internet und Grundfunktionen herkömmlicher Browser vertraut sind, da das Portal eine intuitive Navigation und übersichtliche Darstellung der modellierten Informationen bietet.

5.2 Erweiterung zur Modellierung und Analyse unterer Ebenen

Für einen Modellierer, der sowohl die unteren als auch die oberen Ebenen einer Prozesslandschaft durchgängig werkzeugunterstützt modellieren möchte, ist im Rahmen des Process Landscaping eine Schnittstelle als Erweiterung von *Piazza Palermo* realisiert worden, die die Übertragung von Prozesslandschaften aus *Piazza Palermo* heraus in ein Werkzeug für die Modellierung der unteren Ebenen ermöglicht. Die Software verwendet XML-Dateien zur Speicherung, Verwaltung und Übertragung der Landschaftsdaten. Für eine detaillierte Beschreibung der Datenverwaltung und der der Schnittstelle zugrundeliegenden Softwarearchitektur sei auf [Bro02] verwiesen. Als Modellierungswerkzeug für die unteren Landschaftsebenen, in welches eine Prozesslandschaft übertragen wird, bot sich *Renew* an [Kum00], da dieses mit dem werkzeugeigenen Editor erstellte Prozesslandschaften ebenfalls als XML-Datei speichert und verwaltet. Die XML-Schnittstelle von *Renew* stellt somit einen wichtigen Teil des Bindegliedes zwischen *Piazza Palermo* und *Renew* dar.

Die Handhabung der Schnittstelle wird entlang der in Abschnitt 3.2.1 vorgestellten Vorgehensweise für das Hinzufügen von Ablaufinformationen wiederum am Beispiel der Prozesslandschaft zur komponentenbasierten Softwareentwicklung beschrieben.

Nach dem Öffnen einer zu übertragenden Prozesslandschaft bzw. eines Teilbereiches einer Landschaft (z.B. Verfeinerungen einzelner Aktivitäten) erfolgt der Aufruf der Schnittstellenkomponente in *Piazza Palermo*. Eine Dialogmaske wie in Abbildung 5.4 dargestellt informiert den Anwender über den Fortschritt der Übertragung. Dieser Dialog ist in drei Bereiche unterteilt.

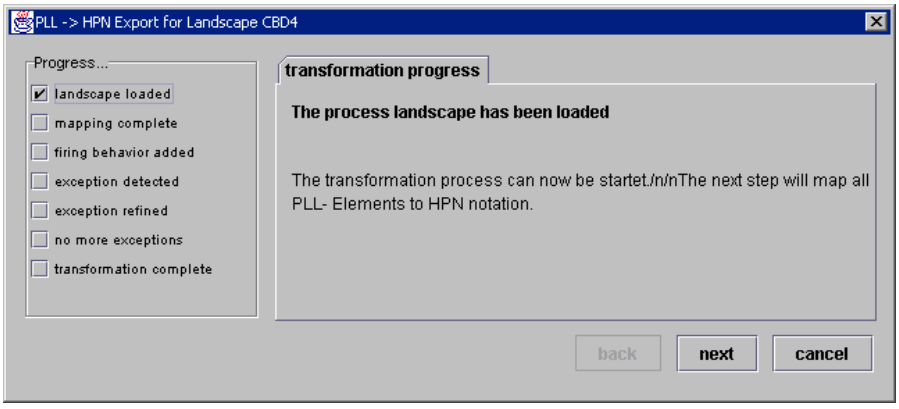


Abbildung 5.4: Ansicht der Schnittstellenoberfläche nach dem Start

Der Bereich *Progress* links in der Abbildung zeigt an, welche Schritte bereits durchgeführt wurden und welche noch folgen. Im unteren Bereich sind die Schaltflächen *back*, *next* und *cancel* zu finden, mit denen der Modellierer den Übertragungsablauf steuern

kann. Im Hauptbereich *transformation progress* findet sich schließlich eine Statusmeldung zum jeweiligen Arbeitsschritt. In einigen Fällen hat der Anwender hier zusätzlich die Möglichkeit, eine Anweisung zum Überspringen des nächsten Arbeitsschrittes zu geben.

Nach dem Start der Schnittstellenkomponente und der Anzeige der in Abbildung 5.4 dargestellten Dialogmaske wird vom Werkzeug zunächst die XML-Struktur für das Hinzufügen von Ablaufinformationen vorbereitet. Dazu sind zuvor alle Verfeinerungen der ausgewählten Prozesslandschaft zu einem zusammenhängenden Petrinetz verknüpft worden, indem unter anderem alle aus syntaktischer Sicht möglichen Datenflüsse (gemäß Definition 3.2.8 in Abschnitt 3.2.1.1) berechnet und dargestellt werden. Nach diesen automatisierten Schritten ist es die Aufgabe des Modellierers, für jede Aktivität das Ein- und Ausgangsschaltverhalten festzulegen.

Abbildung 5.5 zeigt die zugehörige Eingabemaske. Auf der linken Seite wird zunächst eine Aktivität ausgewählt. Für diese werden am unteren Rand weitere Informationen, wie ihre Anordnung im Aktivitätenbaum (als Pfad) und ihre Dokumentation, angezeigt. Auf der rechten Seite der Eingabemaske sind die Schnittstellen im Vor- (*Input interfaces*) und Nachbereich (*Output interfaces*) der ausgewählten Aktivität aufgelistet. Für diese werden bei ihrer Auswahl ebenfalls weitere Informationen (*interface description*) angezeigt. In den Schnittstellenlisten selbst sind zusätzlich Vor- und Nachbereich aller Schnittstellen aufgeführt.

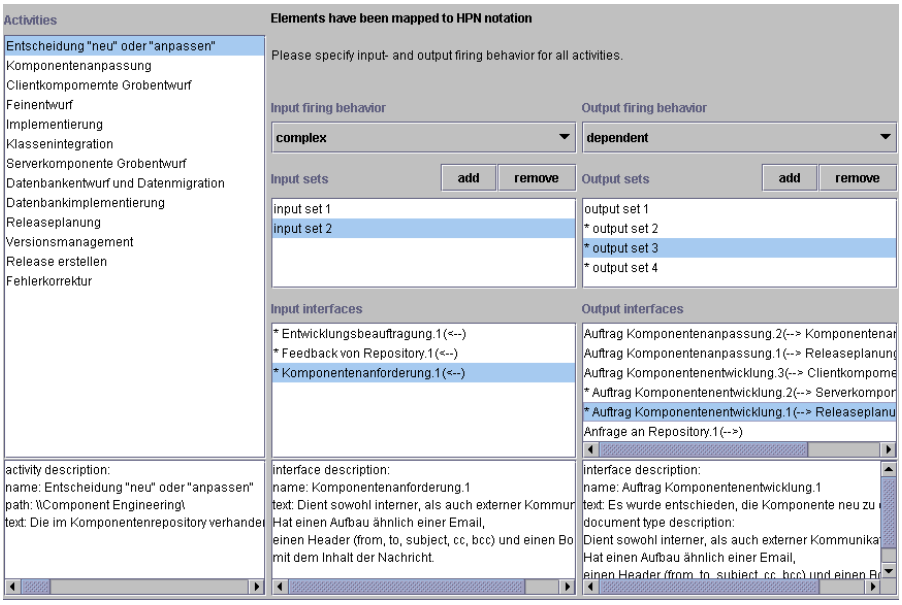


Abbildung 5.5: Festlegen der Schaltverhalten

Ändert der Modellierer das voreingestellte All-Schaltverhalten einer Aktivität, kann er anschließend die Listen der Schalt- (*Input sets*) und Ergebnismengen (*Output sets*)

entsprechend verändern. Das Sternsymbol vor dem Namen einer Schnittstelle zeigt ihm an, dass diese in einer ausgewählten Schalt- bzw. Ergebnismenge enthalten ist.

In Abbildung 5.5 ist die Bearbeitung des Schaltverhaltens der Aktivität *Entscheidung neu oder anpassen* dargestellt. Für diese wurden zwei Schaltmengen *input set* und vier Ergebnismengen *output set* festgelegt. Die Schaltmenge *input set 2* umfasst den gesamten Vorbereich der Aktivität, während die ausgewählte Ergebnismenge *output set 3* lediglich die Schnittstellen *Auftrag Komponentenentwicklung.1* und *Auftrag Komponentenentwicklung.2* enthält. Um den Inhalt der übrigen Schalt- und Ergebnismengen anzuzeigen, kann der Modellierer die jeweiligen Mengen in der Liste auswählen. Die zugehörigen Schnittstellen werden dann unmittelbar mit dem Sternsymbol markiert.

Da alle Informationen, die der Modellierer an dieser Stelle eingibt, direkt in die Prozesslandschaft übertragen werden, kann diese jederzeit in Textform angezeigt werden. Abbildung 5.6 zeigt diese Darstellung.

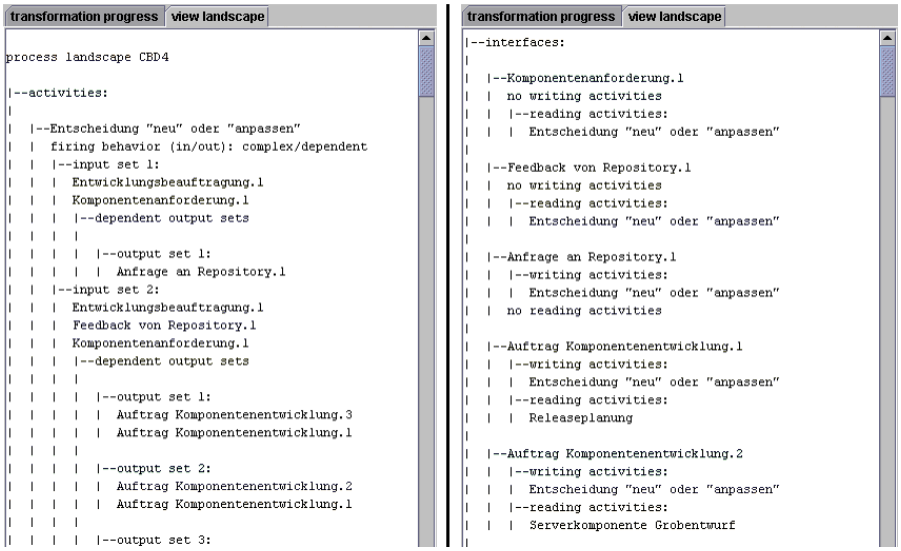


Abbildung 5.6: Darstellung der Prozesslandschaft innerhalb der Schnittstelle

Hier sind für jede Aktivität die zugehörigen Schaltverhalten, Schalt- und Ergebnismengen, gegebenenfalls Relationen zwischen Schalt- und Ergebnismengen (bei abhängigem Ausgangsschaltverhalten, vgl. Definition 3.2.26 in Abschnitt 3.2.1.2) sowie alle Schnittstellen inklusive ihrem Vor- bzw. Nachbereich aufgeführt. Bei Bedarf kann diese textuelle Darstellung über die Zwischenablage in andere Programme übertragen, dort bearbeitet und ausgedruckt werden.

Der Nutzen dieser in Abbildung 5.6 dargestellten Ansicht der Prozesslandschaft besteht in der Gewährleistung eines vollständigen Überblicks über den aktuellen Status der hinzugefügten Ablaufinformationen. Dieser kann bei der Komplexität der Informationen, die pro Aktivität einzugeben sind, ohne diese Ansicht leicht verloren gehen.

Nach dem Festlegen der Schaltverhalten aller Aktivitäten kann der Modellierer optional nach den in Abschnitt 3.2.1.4 definierten Sonderfällen *Zusammengefasste Aktivität*, *Zwei-Phasen-Aktivität* und *Vier-Augen-Prinzip* suchen lassen und diese automatisiert verfeinern. Konnte in der ausgewählten Prozesslandschaft ein Sonderfall identifiziert werden, wird dies mit der in Abbildung 5.7 dargestellten Dialogmaske angezeigt. Im hier dargestellten Fall ist die Aktivität *Anforderungsanalyse erstellen* als Zwei-Phasen-Aktivität identifiziert worden, die für die Erstellung eines Anforderungsdokumentes in der ersten Phase zunächst das Ergebnis der Aktivität *Geschäftsprozess modellieren* anfordert. Die Schnittstelle *Erste Anforderungen.1* wird für beide Phasen benötigt und daher entsprechend dupliziert, bevor die Aktivität *Anforderungsanalyse erstellen* gemäß dem Sonderfall der Zwei-Phasen-Aktivität verfeinert wird (vgl. Diskussion dieses Sonderfalls in Abschnitt 3.2.1.4).

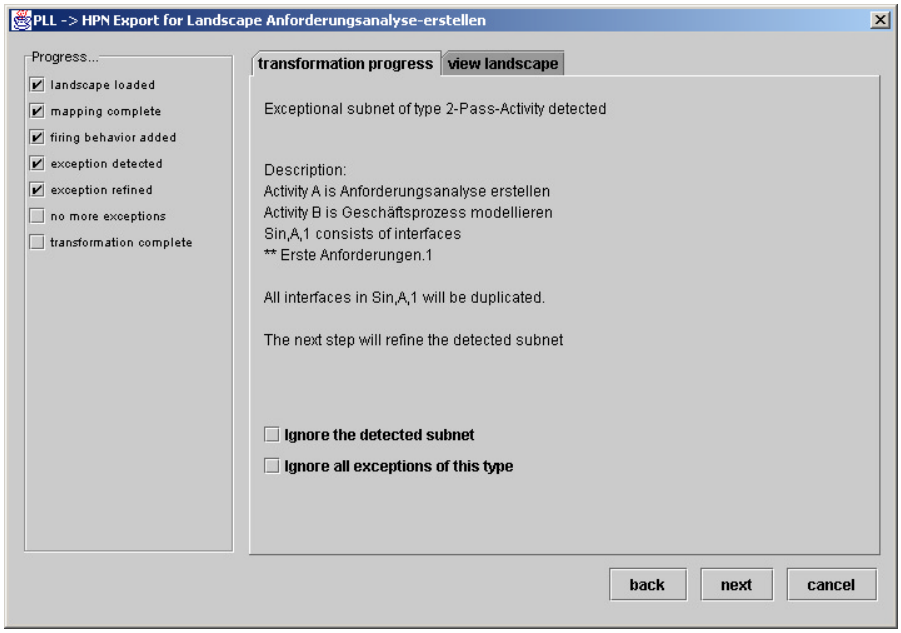


Abbildung 5.7: Ansicht der Schnittstellenoberfläche bei identifiziertem Sonderfall

Die automatisierte Anwendung eines identifizierten Sonderfalls muss vom Modellierer explizit angestoßen werden. Die Durchführung erfolgt dann wie in Abschnitt 3.2.1.4 beschrieben. Die Schnittstellenkomponente ist dabei so realisiert, dass später ohne Probleme weitere Sonderfälle berücksichtigt werden können, die zum jetzigen Zeitpunkt noch nicht spezifiziert sind.

Das Ergebnis einer in *Piazza Palermo* modellierten, mit Ablaufinformationen ergänzten und in *Renew* übertragenen Prozesslandschaft ist ausschnittsweise in Abbildung 5.8 dargestellt. Für die Darstellung in *Renew* wurde jedoch aus Gründen der Übersichtlichkeit von einigen Landschaftsattributen abstrahiert. Lediglich die Namen der Aktivitäten und Schnittstellen sowie die Information, ob es sich um interne oder

externe Schnittstellen handelt, sind angegeben. Damit ist die Übergabe einer Prozesslandschaft von *Piazza Palermo* nach *Renew* inklusive des Hinzufügens von Ablaufinformationen mit Hilfe der Schnittstellenkomponente abgeschlossen. Jetzt kann ein Modellierer die unteren Ebenen dieser Prozesslandschaft beliebig tief und detailliert modellieren.

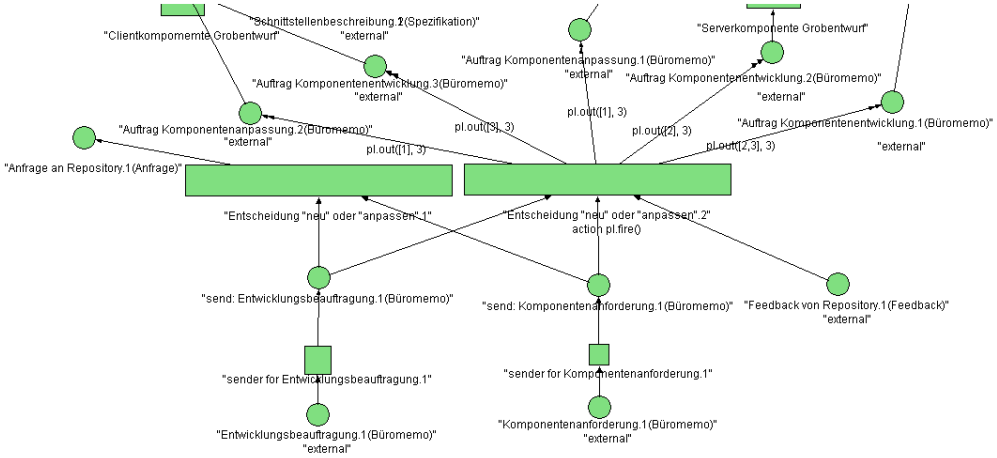


Abbildung 5.8: Ausschnitt aus der in *Renew* importierten Prozesslandschaft

Soll eine Analyse der dynamischen Kommunikationseigenschaften dieser – nun als ein zusammenhängendes, komplexes Petrinetz modellierten – Landschaft erfolgen, bietet es sich an, diese in ihre lokale Sicht umzustrukturieren und in mehrere Teilnetze aufzuteilen, um so die Kommunikation in einer verteilten Prozesslandschaft geeignet betrachten zu können. Für die Analyse dynamischer Kommunikationseigenschaften ist *Renew* um eine Datenbankanbindung und eine Auswertungskomponente erweitert worden. Die Diskussion der Erstellung eines Simulationsmodells und die Auswertung der Simulationsergebnisse ist Gegenstand des nachfolgenden Unterkapitels.

5.3 Modellierung und Simulation der unteren Ebenen einer Prozesslandschaft

Für die Modellierung der unteren Ebenen einer Prozesslandschaft existiert bereits eine Vielfalt von Prozessmodellierungswerkzeugen [CPN00], die auf unterschiedlichen Petrinetz-Varianten basieren. Zur Unterstützung des Process Landscaping ist aus dieser Vielfalt *Renew*, ein am Arbeitsbereich TGI des Fachbereichs Informatik der Universität Hamburg entwickeltes Softwarewerkzeug ausgewählt worden, das den Anforderungen der Methode genügt.

Mit Hilfe des grafischen Editors von *Renew* können Prozesslandschaften von Modellierern mit Petrinetz-Kenntnissen recht schnell und komfortabel modelliert werden.

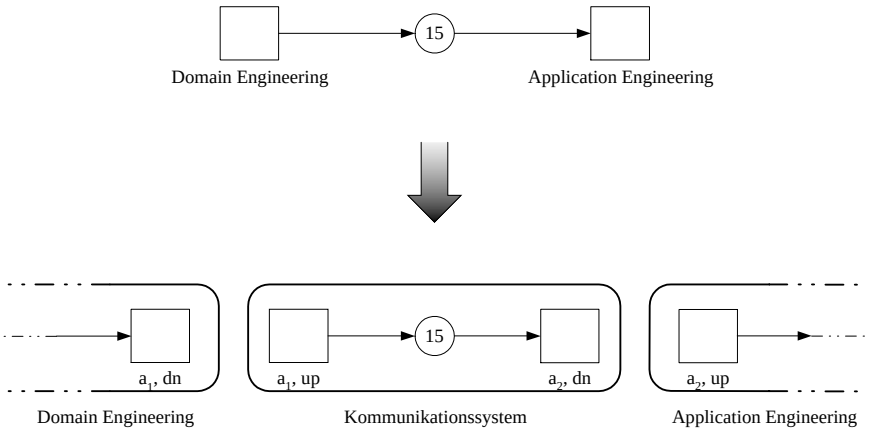
Durch den Einsatz sogenannter Stellvertreterstellen (*virtual places*) kann die grafische Darstellung von Petrinetzen vereinfacht werden. Diese Stellen sind lediglich Kopien der grafischen Darstellung einer Stelle und semantisch von dieser nicht unterscheidbar. Sie verbessern die Lesbarkeit und damit die Verständlichkeit von modellierten Netzen, in denen bestimmte Stellen mit vielen Transitionen verbunden sind. Beispielweise bietet sich die Darstellung der Schnittstelle *Anfrage an Repository.1(Anfrage)* (links in Abbildung 5.8) als Stellvertreterstelle an, nachdem die Prozesslandschaft vollständig mit Ablaufinformationen angereichert und in Renew übertragen ist, da viele Aktivitäten eine entsprechende Anfrage stellen.

In *Renew* wird unter anderem zwischen einfachen und flexiblen Kanten unterschieden. Flexible Kanten haben die Eigenschaft, dass Art und Anzahl derjenigen Marken, die sie transportieren, während der Simulation durch entsprechende Inskriptionen situationsabhängig verändert werden können. In Abbildung 5.8 sind alle fünf von der Aktivität *Entscheidung neu oder anpassen.2* (linke obere Aktivität) nach oben verlaufende Kanten als flexible Kanten modelliert, erkennbar an ihrer doppelten Pfeilspitze. Weitere Kanten und deren Nutzung sind im Handbuch für *Renew* [KWD00] beschrieben.

Für die Simulation einer verteilten Prozesslandschaft zur Analyse dynamischer Kommunikationseigenschaften empfiehlt es sich, diese auf mehrere Petrinetze aufzuteilen. Wie in Abschnitt 4.2.1 bereits für die zur Validation herangezogene Beispielprozesslandschaft beschrieben, wird für ihre lokale Sicht jeder Standort als separates Netz modelliert. Inhaltlich identische Standorte wie beispielsweise *Komponentenentwicklung-1* bis *Komponentenentwicklung-5* sind dabei durch mehrere Instanzen eines Netzes abgebildet. Auch die Kommunikationssysteme werden als separate Netze modelliert. Bei der Modellierung eines Kommunikationskanals innerhalb eines Kommunikationssystems werden die Aktivitäten a_1 und a_2 jeweils durch ein Aktivitätenpaar $(a_{1,dn}, a_{1,up})$ und $(a_{2,dn}, a_{2,up})$ ersetzt. Eine die Kommunikation startende Aktivität a_{dn} wird dabei als **Downlink** bezeichnet, eine zur Kommunikation aufgerufene Aktivität a_{up} als **Uplink**.

Abbildung 5.9 zeigt, wie sich die Aktivitätenpaare sowie die an der Kommunikation beteiligte Schnittstelle auf die drei Teilnetze verteilen. Das Aktivitätenpaar $(a_{1,dn}, a_{1,up})$ repräsentiert die Aktivität *Geschäftsprozess modellieren* als eine der verfeinernden Aktivitäten des *Domain Engineering*. Die beiden Elemente $a_{1,dn}$ und $a_{1,up}$ sind auf die Teilnetze *Domain Engineering* und *Kommunikationssystem* verteilt. Analog finden sich die Elemente des Aktivitätenpaares $(a_{2,dn}, a_{2,up})$ in den Teilnetzen *Kommunikationssystem* und *Application Engineering* wieder.

Die verwendeten Mechanismen zur Kommunikation der verschiedenen Netze untereinander werden in *Renew* mit Hilfe der Programmiersprache Java implementiert. Diese Umsetzung wird nachfolgend für einige Situationen innerhalb der Beispielprozesslandschaft zur komponentenbasierten Softwareentwicklung erläutert, wie sie zur Validation der Analysemethode des Process Landscaping durchgeführt worden ist.



15 Geschäftsprozessmodell
a_1 Geschäftsprozessmodell modellieren
a_2 Anforderungsanalyse erstellen

Abbildung 5.9: Kommunikationskanal innerhalb eines Kommunikationssystems

Ein Dokumenttyp kann in Java wie folgt als Attributmenge einer Klasse formuliert werden:

```
public class Token{
    private int complexity;
    private NetInstance sender;
    private NetInstance via;
    private NetInstance receiver;
    private double volume;
}
```

`NetInstance` repräsentiert dabei eine Basisklasse aller modellierten Teilbereiche einer Prozesslandschaft.

Die Modellierung der Responderstelle im Netz einer sendenden Aktivität stellt eine netzübergreifende Besonderheit dar. Abbildung 5.10 zeigt die zugehörigen Ausschnitte der Petrinetze am Beispiel des Application Engineering am Standort *Anwendungsentwicklung*, das mit dem Projektmanagement und dem zentralen Qualitätsmanagement kommuniziert. Die Kommunikationsbereitschaft wird potentiellen Initiatoren durch das Belegen der Responderstelle mit einer Marke angezeigt. Durch die Aufteilung der Prozesslandschaft auf verschiedene Petrinetze für verschiedene Standorte sind Initiator und Responder jedoch auf unterschiedliche Netze aufgeteilt, obwohl beide Stellen zum Vorbereich der sendenden Aktivität gehören. Daher wird der erfolglose Kommunikationsversuch eines Initiators mit einem bereits beschäftigten Responder

jetzt dadurch verhindert, dass letzterer jedem potentiellen Initiator immer eine entsprechende Zustandsänderung der Responderstelle mitteilt. So wird ein Kommunikationsversuch mit einem beschäftigten Responder gar nicht erst initiiert.

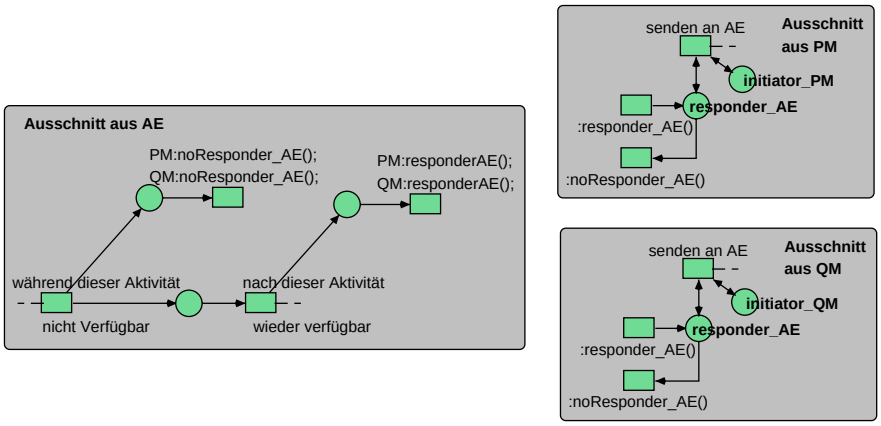


Abbildung 5.10: Modellierung einer Responderstelle innerhalb eines Petrinetzes

Die links im Petrinetz *Anwendungsentwicklung* dargestellte Aktivität entfernt eine Marke von der Responderstelle des Netzes. Sie stellt das Gegenstück zu den Aktivitäten aller mit dem Netz kommunizierenden Initiatoren dar. Beim Schalten dieser Aktivitäten wird eine Marke auf den Stellen *responder_AE* in den zugehörigen Netzen (rechts in Abbildung 5.10) entfernt. Sowohl Responder- als auch Initiatorstellen sind über bidirektionale Kanten mit der sendenden Aktivität verbunden. Für die Initiatorstellen *initiator_PM* und *initiator_QM* ist so eine permanente Kommunikationsbereitschaft gewährleistet. Für die Responderstelle, die außerhalb des Senders modelliert ist, ist über diesen Mechanismus ebenfalls die Anzeige der Kommunikationsbereitschaft anzeigbar.

Diese Form der Modellierung ist lediglich für das zentrale Qualitätsmanagement verändert worden. Dort sind die Responderstellen als permanent belegt modelliert, um mögliche Verklemmungen des gesamten Simulationsmodells zu verhindern. Eine permanente Belegung ermöglich zwar, dass das Application Engineering ein weiteres Anforderungsdokument erstellen kann, während ein zuvor erstelltes vom Qualitätsmanagement geprüft wird. Das Review dieses weiteren Dokumentes kann so aber erst vom Application Engineering angestoßen werden, wenn das Reviewergebnis des ersten Dokuments zurückgeschickt worden ist und das Application Engineering somit nicht mehr auf das erste Reviewergebnis wartet.

Parallele Modellierung durch ein Team von Modellierern ist im Gegensatz zu *Piazza Palermo* bei *Renew* nicht vorgesehen. Auch eine getrennte Nutzung durch Modellierer und Prozessbeteiligte ist nicht möglich. Beides ist allerdings auch von Anfang an nicht Ziel des Werkzeugs gewesen. Einen Schwerpunkt des Werkzeugs bildet dagegen die Möglichkeit der Simulation von Prozesslandschaften. Die Simulationsläufe wer-

den von einer Simulationskomponente ausgeführt, die ebenfalls von *Renew* zur Verfügung gestellt wird. Diese Simulationsläufe können sowohl animiert als auch nicht animiert durchgeführt werden. Da für die Simulation einer Prozesslandschaft im Rahmen des Process Landscaping Massendaten anfallen, sollte die Animation lediglich für die Überprüfung der korrekten Modellierung der einzelnen Teilnetze eingesetzt werden; auf eine Animation der eigentlichen Simulationsläufe sollte jedoch aus Performanzgründen verzichtet werden.

Die Ergebnisse der verschiedenen Simulationsläufe werden persistent in einer Datenbank gespeichert, deren Schnittstelle zu *Renew* im Rahmen des Process Landscaping realisiert wurde. Weitere zur Bewertung der Kommunikationseffizienz einer Prozesslandschaft notwendigen Daten sind in einer XML-Datei gespeichert. Diese im Folgenden als Projektdatei bezeichnete Datei beinhaltet beispielsweise Informationen über verwendete Kostenmodelle und Relationen zwischen den einzelnen Netzen.

Eine erste Auswertung der Simulationsergebnisse im Zusammenhang mit den in der Projektdatei gespeicherten Daten wird von einer Auswertungskomponente durchgeführt, die ebenfalls im Rahmen des Process Landscaping als Erweiterung von *Renew* realisiert worden ist. Diese Auswertungskomponente stellt vor allem verschiedene Process Landscaping-spezifische Analysealgorithmen zur Verfügung. Im Einzelnen sind dies Algorithmen

- zur Berechnung von Prozessauslastungen,
- zur Bestimmung von Pfadlängen,
- zur Ermittlung von Initiator-/Responderraten und
- zur Aufbereitung der stochastischen Auswertung.

Das Ergebnis der verschiedenen Algorithmen wird in einem dreigeteilten Fenster dargestellt, das in Abbildung 5.11 zu sehen ist. Im linken oberen Bereich ist die hierarchische Datenstruktur der Simulationsläufe abgebildet, die die Gliederung in Projekte, Experimente und Netze zeigt. Wird ein Eintrag selektiert, werden im rechten oberen Bereich weitere Details angezeigt. In Abbildung 5.11 bestehen diese Details aus den Kommunikationskosten, durchschnittlichen Pfadlängen und Prozessauslastungen für das Projektmanagement eines Softwareentwicklungsprojektes, welches als separates Petrinetz, im Beispiel als *Cluster PM* bezeichnet, modelliert ist. Die dort berechneten Werte basieren auf zwei Simulationsläufen. Ihre Darstellung erfolgt textuell, indem zunächst die Anzahl der Stichproben, ein erwartungstreuer Mittelwert und ein erwartungstreuer Schätzer der Varianz aufgeführt werden. Darunter ist die Verteilung der Stichprobenwerte aufgelistet, indem in jeder Zeile ein halboffenes Intervall mit relativer Häufigkeit ausgegeben wird. Im unteren Bereich des Fensters in Abbildung 5.11 werden während der Auswertung Logging-Informationen ausgegeben wie beispielsweise das erfolgreiche Öffnen der Datenbank oder das Einlesen der darin enthaltenen Daten.

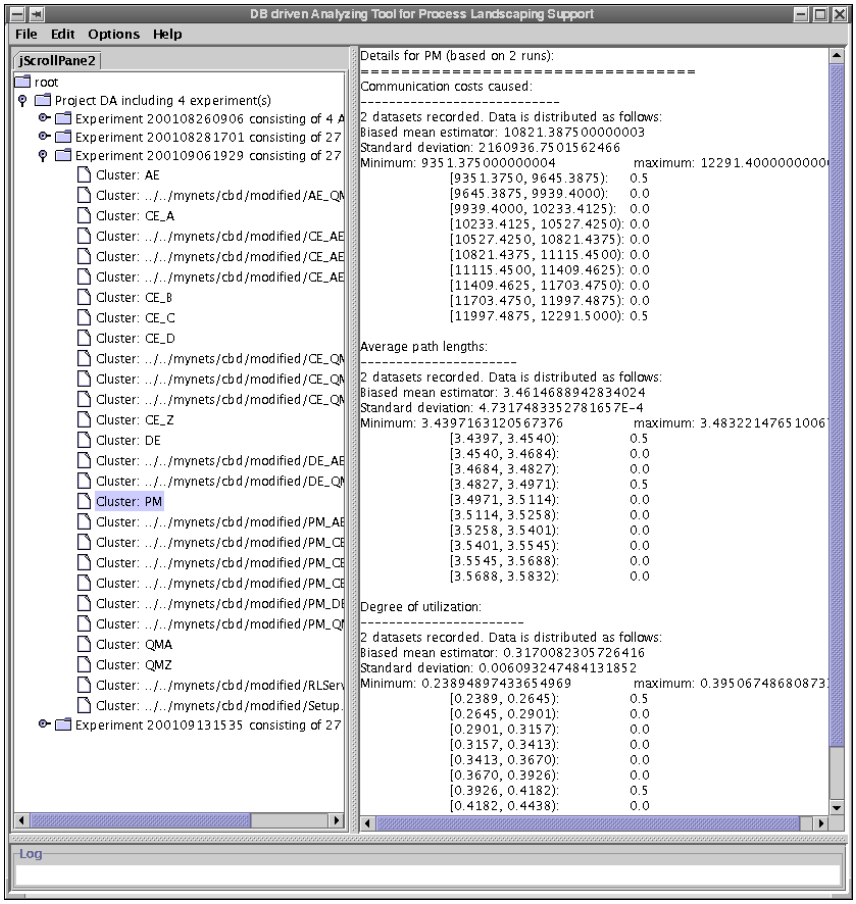


Abbildung 5.11: Oberfläche der Auswertungskomponente von *Renew*

Neben der Berechnung und Darstellung der Auswertungsdaten bietet die Auswertungskomponente weitergehende Funktionalitäten, die eine Arbeitserleichterung für den Modellierer darstellen. Beispielsweise kann die Funktionsweise der Auswertungskomponente über Parameter beeinflusst werden, die sich auf datenbankbezogene Einstellungen zur Datenbankadresse, Benutzername und Passwort beziehen. Diese Werte überschreiben bzw. ergänzen Angaben in der Projektdatei. Auf diese Weise wird der Austausch von Projektdateien ermöglicht. Weitere Einstellmöglichkeiten sind die Anzahl standardmäßig durchzuführender Experimente (Mengen von Simulationsläufen, vgl. Abbildung 4.3 in Abschnitt 4.2.2.1) und die Ein- bzw. Ausschaltung der animierten Simulation.

5.4 Status der Werkzeugunterstützung

Mit der in den vorangegangenen Abschnitten vorgestellten Software existiert eine Werkzeugunterstützung, die die Modellierungsschritte des Process Landscaping – bis auf die Umstrukturierung von der logischen in die lokale Sicht – vollständig unterstützt. Für die Modellierung der oberen Ebenen einer Prozesslandschaft ist dabei zusätzlich besonderer Wert auf das parallele Arbeiten in einem Modellierungsteam gelegt worden, welches zudem noch geografisch verteilt arbeiten kann. Dass die Umstrukturierung einer gegebenen Prozesslandschaft werkzeuggestützt durchgeführt werden kann, ist jedoch zumindest für die oberen Ebenen prototypisch in [Poh00] gezeigt worden. Eine durchgängige Modellierung von den oberen zu den unteren Ebenen ist durch die in Abschnitt 5.2 dargestellte Schnittstellenkomponente ebenfalls ermöglicht. Sie erlaubt einen Werkzeugwechsel ohne Bruch in der Anwendung des Process Landscaping.

Die werkzeuggestützte Analyse konzentriert sich für die oberen Ebenen auf Konsistenzprüfungen einer erstellten Prozesslandschaft. Die Möglichkeit der Festlegung von Attributwerten für eine Kommunikationsinfrastruktur ist ebenfalls gegeben. Die Auswertung der Attributwerte erfolgt zurzeit noch prototypisch über deren Export in eine Datenbank (MS-Access) und anschließende SQL-Abfragen.

Für die unteren Ebenen ist eine vollständige Werkzeugunterstützung vorhanden. Sie ermöglicht nicht nur die Simulation und damit die Analyse dynamischen Verhaltens einer Prozesslandschaft, sondern bietet gleichzeitig die Auswertung der gewonnenen Simulationsergebnisse für die Process Landscaping-spezifischen Kommunikationseigenschaften an.

Kapitel 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung der Arbeit

Die Methode des Process Landscaping unterstützt ein strukturiertes Vorgehen zur Erstellung und Analyse einer verteilten Prozesslandschaft. Sie berücksichtigt verschiedene Aspekte zu Vorgehen, Rollen, Ergebnissen und unterstützenden Techniken, die in der Literatur auch als Komponenten einer Methode bezeichnet werden [Hey95, HB96]. Für jeden Vorgehensschritt sind verantwortliche bzw. beteiligte Rollen sowie die Zielgruppe der Ergebnisse benannt. Ergebnisse der Methode sind neben einer Prozesslandschaft das Erreichen konkreter Analyseziele für verschiedene Abstraktionsebenen und zugehörige Interpretationsrahmen. Das Ergebnis der Modellierungsschritte, die Prozesslandschaft, ist unter Berücksichtigung sowohl allgemeiner Prinzipien wie den Grundsätzen ordnungsmäßiger Modellierung als auch methodenspezifischer Grundsätze wie beispielsweise der besonderen Schnittstellenbetrachtung entstanden. Aber auch das Erreichen der Analyseziele – Bewertung und ggf. Verbesserung von Kommunikationseigenschaften unter besonderer Berücksichtigung von Verteilungsaspekten – wird durch methodenneutrale und -spezifische Grundsätze unterstützt. Schließlich stehen für jeden Methodenschritt – soweit sinnvoll – Techniken zur Erreichung der verschiedenen Ergebnisse zur Verfügung. Diese reichen von unterschiedlichen Arten der Wissensakquise für die Modellierung einer Prozesslandschaft über die Umstrukturierung in die lokale Sicht bis hin zu Vorgehensweisen für die Bewertung von quantitativen Analyseergebnissen.

Das Process Landscaping unterstützt im Gegensatz zu vielen anderen Modellierungs- und Analyseansätzen auch unstrukturierte Prozesse, für die kein fester Ablauf vorgegeben ist. Dazu wird sowohl die Möglichkeit der Modellierung ohne Ablaufinformationen gegeben, und dies für verschiedene Abstraktionsebenen, als auch die Möglichkeit der Analyse ihrer statischen Kommunikationsinfrastruktur. Ein konsistenter Modellierungsübergang zu Prozessen mit Ablaufinformationen wird durch eine entsprechende konstruktive Erweiterung der zugrundeliegenden Petrinetze gewährleistet. Dieser Übergang zeichnet die Modellierungsschritte des Process Landscaping zusammen mit seinen Alleinstellungsmerkmalen – der expliziten Schnittstellenbetrachtung und der Umstrukturierungsmöglichkeit von der logischen in die lokale Sicht einer Prozess-

landschaft – aus. Für strukturierte Prozesse bietet die vorgestellte Methode die Möglichkeit, über die Analyse des dynamischen Kommunikationsverhaltens unter besonderer Berücksichtigung von Verteilungsaspekten die Kommunikationseffizienz einer Prozesslandschaft als ein Kriterium für Optimierungsansätze zu untersuchen.

Die formale Basis einer Prozesslandschaft bedient sich der Eigenschaften verschiedener Petrinetz-Varianten. Ausgehend von der Basisdefinition einer Prozesslandschaft wird diese schrittweise um Abbildungen und Attribute erweitert, die zur Modellierung und Analyse der verschiedenen Abstraktionsebenen erforderlich sind. Dabei sind teilweise Definitionen aus der Literatur übernommen, teilweise sind die definierten Begriffe – wie beispielsweise das Schaltverhalten – jedoch anders eingeführt. Für die Analyse verschiedener Kommunikationseigenschaften existieren zusätzliche, landschaftsspezifische Attribute und Abbildungen.

Ein wichtiges Merkmal der formalen Grundlage des Process Landscaping ist die Formulierung der Basis einer Prozesslandschaft und aller vier Ausbaustufen als Kombination eines gerichteten Baumes zusammen mit einem je nach Ausbaustufe unterschiedlich ausgeprägten Petrinetz. Während der Aktivitätenbaum die Gesamtübersicht über die hierarchische Struktur einer modellierten Prozesslandschaft ermöglicht, unterstützt das Petrinetz die Betrachtung vorhandener Aktivitäten und ihrer Zugriffe auf erzeugte bzw. genutzte Informationsobjekte auf unterschiedlichen Abstraktionsebenen. Die vorgestellten Verfeinerungskonzepte sowohl für Aktivitäten als auch für Schnittstellen unterscheiden sich von der traditionellen Vorgehensweise der in der Literatur vorgeschlagenen Konzepte, da sie auch die Umgebung zu verfeinernder Knoten eines Petrinetzes einbeziehen. Dies erfordert zwar die Einhaltung einer festen Reihenfolge bei der Durchführung von Verfeinerungen, ermöglicht aber trotzdem ein sukzessives Hinzufügen von Detailinformationen überall dort, wo es aus Sicht des Modellierers sinnvoll erscheint.

Bei den oberen Ebenen einer Prozesslandschaft wird durch ihre Modellierung mit PLL explizit von Ablaufinformationen abstrahiert, nicht aber von der Schnittstellenbetrachtung zwischen Aktivitäten. Ein konsistenter Modellierungsübergang zu den unteren Ebenen einer Prozesslandschaft wird durch die konstruktive Erstellung einer kohärenten Prozesslandschaft ermöglicht. Dort wird eine explizite Schnittstellenbetrachtung insbesondere durch die Unterscheidung in interne und externe Schnittstellen unterstützt. Sie ermöglicht vor allem in der lokalen Sicht eine fokussierte Betrachtung der Kommunikation zwischen geografisch verteilt arbeitenden Aktivitäten.

Insgesamt unterstützt so die Notation einer Prozesslandschaft als Aktivitätenbaum zusammen mit einem Petrinetz alle in Abschnitt 1.1 diskutierten Problemstellungen, indem sie für die in Kapitel 2 präsentierten Lösungsmöglichkeiten eine geeignete formale Basis anbietet. Sie ist insbesondere notwendige Voraussetzung für die Analyse des Process Landscaping, die sich in die Analyse statischer und dynamischer Kommunikationseigenschaften unterteilt: Die Analyse statischer Kommunikationseigenschaften setzt den Fokus auf die Kommunikationsinfrastruktur einer Prozesslandschaft. Sie hat die Minimierung eines Kommunikationskostenfaktors für diese Infrastruktur bezüglich verschiedener Strategien zum Ziel. Die Analyse dynamischer Kommunikationsei-

genschaften untersucht mit Hilfe der Simulation einer verteilten Prozesslandschaft deren Kommunikationseffizienz mit dem Ziel ihrer Optimierung durch die Verbesserung sowohl des Auslastungsgrades von Aktivitäten, deren geografischer Verteilung, der Informationsverteilung als auch des Informationsflusses. Beide Analyseformen sind bislang nicht in der Literatur diskutiert worden, bieten jedoch für fachlich sinnvoll strukturierte Prozesse ein geeignetes Hilfsmittel für deren Effizienzsteigerung.

Die für die Methode vorhandene Werkzeugunterstützung erlaubt die Modellierung aller Abstraktionsebenen mit und ohne Berücksichtigung von Ablaufinformationen. Für die oberen Ebenen ist dazu ein Modellierungswerkzeug entwickelt worden; für die unteren Ebenen ist eines von vielen bereits vorhandenen Petrinetz-Werkzeugen ausgewählt worden. Eine XML-Schnittstelle ermöglicht einen konsistenten Übergang von den oberen zu den unteren Ebenen einer Prozesslandschaft. Diese durchgängige Modellierungsunterstützung verbessert die Einsatzmöglichkeit des Process Landscaping insbesondere für komplexe Prozesslandschaften und trägt somit auch zur Steigerung der Akzeptanz der Methode bei, wie dies allgemein bei werkzeuggestützten Methoden der Fall ist [LSS99]. Bezogen auf die Analyseschritte des Process Landscaping wird die Untersuchung der Kommunikationseffizienz erst durch eine geeignete Werkzeugunterstützung möglich. Dies ist mit der Erweiterung des Petrinetz-Werkzeugs *Renew* durch die Anbindung an eine Datenbank zur konsistenten Speicherung der Simulationsergebnisse und deren Verarbeitung durch eine Auswertungskomponente gegeben.

6.2 Ausblick auf weiterführende Arbeiten

Weiterführende Arbeiten im Rahmen des Process Landscaping konzentrieren sich auf drei verschiedene Aspekte, konkret

- der Verbesserung und Erweiterung der Werkzeugunterstützung,
- der Erweiterung von Analysemöglichkeiten verteilter Prozesslandschaften und
- der Anwendung der Methode in weiteren Projekten.

Bezüglich der Werkzeugunterstützung ist unter anderem eine Optimierung von Teilen der Benutzeroberfläche und die Realisierung von Schnittstellen zwischen *Piazza Palermo* und weiteren Petrinetz-basierten Softwarewerkzeugen insbesondere für den Übergang zwischen den oberen und unteren Ebenen einer Prozesslandschaft sinnvoll. Die bereits Ende des Abschnitts 5.1.1 angesprochene Realisierung zur verbesserten werkzeuggestützten Analyse der Kommunikationsinfrastruktur einer Prozesslandschaft ist ebenfalls ein zukünftiges Ziel. Vor allem erscheint jedoch eine Unterstützung für die Umstrukturierung von der logischen in die lokale Sicht hilfreich. Die im Rahmen dieser Arbeit noch manuell durchgeführte Umstrukturierung ist relativ zeitaufwändig. Weiterführende Ansätze bezüglich des Umstrukturierungskonzeptes werden in [KW02] vorgestellt, bei denen eine rollenbasierte in die logische Sicht umstrukturiert wird. Diese Idee basiert auf dem gleichen Konzept wie die Umstrukturierung von der logischen in die lokale Sicht.

Die Analysemöglichkeiten des Process Landscaping sind bislang hauptsächlich im Rahmen dieser Arbeit – am Beispiel der komponentenbasierten Softwareentwicklung – validiert worden. Empirische Untersuchungen in realen Projekten können die hier diskutierte Nutzenbetrachtung weiter untermauern und die Hinzunahme der Kommunikation als Entscheidungskriterium bei Prozessoptimierungen zum Regelfall werden lassen.

Verteilte Prozesslandschaften bieten eine Vielzahl von Analysemöglichkeiten weiterer Aspekte, die bislang nicht oder nur unzureichend berücksichtigt worden sind. Beispielsweise bietet es sich gerade für geografisch verteilte Prozesse an, deren Autonomie bezüglich der von ihnen benötigten und/oder erstellten Daten oder bezüglich der Durchführung ihrer Aufgaben zu betrachten und im Gesamtzusammenhang aller kooperierenden Prozesse zu bewerten. Erste Ideen zur Autonomiebetrachtung in einer verteilten Softwareprozesslandschaft werden in [GW02] diskutiert.

Process Landscaping ist bereits auf komplexe Prozesse in der Telekommunikationsbranche [GW00a] sowie auf Softwareprozesse zur komponentenbasierten Softwareentwicklung [GW00b] und auch zur Erstellung multimedialer Lehr- und Lernsysteme [KW02] angewandt worden. Im Bereich der Telekommunikation diente die Modellierung der Darstellung und Schulung neu zu konzipierender Geschäftsprozesse mit besonderem Schwerpunkt auf der Festlegung von Schnittstellen zwischen den Prozessen und Aktivitäten, die oft über mehrere organisatorische Geschäftseinheiten hinweg verliefen. Letztere waren zusätzlich meist geografisch über den gesamten Globus verteilt. Die Modellierung verteilter Softwareprozesse zur komponentenbasierten Softwareentwicklung in [GW00b] hatte die explizite Formulierung der einzelnen Modellierungsschritte des Process Landscaping und die Einordnung der Methode in den wissenschaftlichen Hintergrund zum Ziel. Schließlich stand bei der Modellierung einer Prozesslandschaft zur Erstellung multimedialer Lehr- und Lernsysteme die Umstrukturierung einer rollenbasierten Sicht in eine logische Sicht unter Anwendung des Umstrukturierungskonzeptes der Methode im Vordergrund der Betrachtung.

Die Erfahrungen in allen drei beschriebenen Projekten haben nicht nur gezeigt, dass Handlungsbedarf für die in Abschnitt 1.1 diskutierten Problemstellungen besteht, insbesondere für die explizite Berücksichtigung von Schnittstellen innerhalb von und zwischen Prozessen, der Lösungsansatz des Process Landscaping hat sich dabei auch als geeignet erwiesen. Die Anwendung des Process Landscaping in weiteren Projekten kann belegen, dass die Methode nicht nur zur Modellierung und Analyse von Softwareprozessen geeignet ist, sondern auch für andere Formen von Geschäftsprozessen. Dies ist zwar im Bereich der Telekommunikation schon geschehen, sollte aber durch Erfahrungswerte weiterer Projekte detailliert nachgewiesen werden.

Insgesamt bleibt festzuhalten, dass der Einsatz des Process Landscaping in weiteren Prozessmodellierungs- und -analyseprojekten den im Rahmen dieser Arbeit erbrachten Nachweis des Nutzens auch in der täglichen Praxis bestätigen kann und daher für verschiedene prozessorientierte Projekte angestrebt wird.

Anhang A

Aufbau der Beispielprozesslandschaft

Der Anhang enthält viele Detailinformationen der Prozesslandschaft, die die komponentenbasierte Softwareentwicklung darstellt. Diese dient als Beispiel sowohl zur Erläuterung der Methode des Process Landscaping als auch als Validationsgrundlage der Analyse. Der Anhang ist unterteilt in Informationen über die Aktivitäten der Prozesslandschaft ω und ihrer Verteilung auf die verschiedenen Standorte (Abschnitt A.1) und in Detailinformationen zur statischen (Abschnitt A.2) und zur dynamischen Analyse (Abschnitt A.3). Die beiden letztgenannten Unterkapitel zeigen neben den konkreten Werten der instanziierten Prozesslandschaft zusätzlich Vorberechnungen, die die Grundlage der in Kapitel 4 diskutierten Optimierungsansätze bilden.

A.1 Prozesslandschaft ω

Aktivität	**Sublevel**	**Toplevel**
Anforderungsanalyse erstellen	Kundenlokation	Zentrale
Abnahme mit dem Kunden	Kundenlokation	
Kommunikation mit dem Kunden	Kundenlokation	
Release erstellen	Anwendungsentwicklung	Zentrale
Releaseplanung	Anwendungsentwicklung	
Versionsmanagement	Anwendungsentwicklung	
Spezifikation erstellen	Anwendungsentwicklung	
System installieren	Anwendungsentwicklung	
System integrieren	Anwendungsentwicklung	
Systemarchitektur erstellen	Anwendungsentwicklung	
Zur Abnahme freigeben	Anwendungsentwicklung	
Komponente anfordern	Anwendungsentwicklung	
Komponente hinzufügen	Anwendungsentwicklung	

Aktivität	**Sublevel**	**Toplevel**
Technische Komponentenanforderung erstellen	Anwendungsentwicklung	
Vorhandene Komponenten auswählen	Anwendungsentwicklung	
Komponente kaufen	Anwendungsentwicklung	
Feedback an DE geben	Anwendungsentwicklung	
Clientkomponente Grobentwurf	Komponentenentwicklung-1	Zentrale
Datenbankentwurf und Datenmigration	Komponentenentwicklung-1	
Datenbankimplementierung	Komponentenentwicklung-1	
Feinentwurf	Komponentenentwicklung-1	
Implementierung	Komponentenentwicklung-1	
Klassenintegration	Komponentenentwicklung-1	
Serverkomponente Grobentwurf	Komponentenentwicklung-1	
Komponentenanpassung	Komponentenentwicklung-1	
Release erstellen (CE)	Komponentenentwicklung-1	
Releaseplanung (CE)	Komponentenentwicklung-1	
Versionsmanagement(CE)	Komponentenentwicklung-1	
Fehlerkorrektur	Komponentenentwicklung-1	
Richtlinien anfragen	Komponentenentwicklung-1	
Entscheidung neu oder anpassen	Managementlokation	Zentrale
Geschäftsprozess modellieren	Managementlokation	
Wiederverwendungsrepository verwalten	Managementlokation	
Projekt einschätzen	Managementlokation	
Buy or Build-Entscheidung treffen	Managementlokation	
CE mit Komponentenentwicklung beauftragen	Managementlokation	
Markt analysieren	Managementlokation	
Projekt initialisieren	Managementlokation	
Projektabschluss	Managementlokation	
Projektdarstellung nach außen	Managementlokation	
Projektplanerstellung	Managementlokation	
Projektsteuerung und -kontrolle	Managementlokation	
Ressourcenplanung	Managementlokation	
Risikomanagement	Managementlokation	
Teamzusammenstellung	Managementlokation	
Testablaufverfolgung	Managementlokation	
Richtlinienerweiterung weiterleiten	Managementlokation	
Richtlinienerstellung	Qualitätsmanagement-Z	Zentrale
Testplanerstellung	Qualitätsmanagement-Z	
Reviewauswertung	Qualitätsmanagement-Z	
Testauswertung	Qualitätsmanagement-Z	
Reviewdurchführung	Qualitätsmanagement-Z	
Testdurchführung	Qualitätsmanagement-Z	
Testvorbereitung	Qualitätsmanagement-Z	

Aktivität	**Sublevel**	**Toplevel**
Clientkomponente Grobentwurf	Komponentenentwicklung-2	Außenstelle-1
Datenbankentwurf und Datenmigration	Komponentenentwicklung-2	
Datenbankimplementierung	Komponentenentwicklung-2	
Feinentwurf	Komponentenentwicklung-2	
Implementierung	Komponentenentwicklung-2	
Klassenintegration	Komponentenentwicklung-2	
Serverkomponente Grobentwurf	Komponentenentwicklung-2	
Komponentenanpassung	Komponentenentwicklung-2	
Release erstellen (CE)	Komponentenentwicklung-2	
Releaseplanung (CE)	Komponentenentwicklung-2	
Versionsmanagement (CE)	Komponentenentwicklung-2	
Fehlerkorrektur	Komponentenentwicklung-2	
Richtlinien anfragen	Komponentenentwicklung-2	
Testauswertung	Qualitätsmanagement-A	Außenstelle-1
Testplanerstellung	Qualitätsmanagement-A	
Testablaufverfolgung	Qualitätsmanagement-A	
Reviewauswertung	Qualitätsmanagement-A	
Richtlinienerweiterung vorschlagen	Qualitätsmanagement-A	
Reviewdurchführung	Qualitätsmanagement-A	
Testdurchführung	Qualitätsmanagement-A	
Testvorbereitung	Qualitätsmanagement-A	
Clientkomponente Grobentwurf	Komponentenentwicklung-3	Außenstelle-2
Datenbankentwurf und Datenmigration	Komponentenentwicklung-3	
Datenbankimplementierung	Komponentenentwicklung-3	
Feinentwurf	Komponentenentwicklung-3	
Implementierung	Komponentenentwicklung-3	
Klassenintegration	Komponentenentwicklung-3	
Serverkomponente Grobentwurf	Komponentenentwicklung-3	
Komponentenanpassung	Komponentenentwicklung-3	
Release erstellen (CE)	Komponentenentwicklung-3	
Releaseplanung (CE)	Komponentenentwicklung-3	
Versionsmanagement (CE)	Komponentenentwicklung-3	
Fehlerkorrektur	Komponentenentwicklung-3	
Richtlinien anfragen	Komponentenentwicklung-3	
Clientkomponente Grobentwurf	Komponentenentwicklung-4	Außenstelle-3
Datenbankentwurf und Datenmigration	Komponentenentwicklung-4	
Datenbankimplementierung	Komponentenentwicklung-4	
Feinentwurf	Komponentenentwicklung-4	
Implementierung	Komponentenentwicklung-4	
Klassenintegration	Komponentenentwicklung-4	
Serverkomponente Grobentwurf	Komponentenentwicklung-4	

Aktivität	**Sublevel**	**Toplevel**
Komponentenanpassung	Komponentenentwicklung-4	
Release erstelle (CE)	Komponentenentwicklung-4	
Releaseplanung (CE)	Komponentenentwicklung-4	
Versionsmanagement (CE)	Komponentenentwicklung-4	
Fehlerkorrektur	Komponentenentwicklung-4	
Richtlinien anfragen	Komponentenentwicklung-4	
Clientkomponente Grobentwurf	Komponentenentwicklung-5	Außenstelle-4
Datenbankentwurf und Datenmigration	Komponentenentwicklung-5	
Datenimplementierung	Komponentenentwicklung-5	
Feinentwurf	Komponentenentwicklung-5	
Implementierung	Komponentenentwicklung-5	
Klassenintegration	Komponentenentwicklung-5	
Serverkomponente Grobentwurf	Komponentenentwicklung-5	
Komponentenanpassung	Komponentenentwicklung-5	
Release erstellen (CE)	Komponentenentwicklung-5	
Releaseplanung (CE)	Komponentenentwicklung-5	
Versionsmanagement (CE)	Komponentenentwicklung-5	
Fehlerkorrektur	Komponentenentwicklung-5	
Richtlinien anfragen	Komponentenentwicklung-5	

Tabelle A.1: Aktivitäten der Prozesslandschaft ω und ihre lokale Verteilung

A.2 Detailinformationen zur statischen Analyse

In diesem Abschnitt sind alle zur Berechnung der in Abschnitt 2.3.2 diskutierten Kommunikationskostenfaktoren Kkf notwendigen Kommunikationskanäle in Tabellenform dargestellt. Im Einzelnen ist dies eine Liste aller existierenden Kommunikationskanäle zusammen mit ihren Attributen, Nutzungshäufigkeiten und Kosten (Tabellen A.2 bis A.18).

In den Tabellen gehören jeweils drei Zeilen zur Beschreibung eines Kommunikationskanals. In der ersten Spalte bezeichnet die erste Zeile dabei jeweils die sendende Aktivität, die zweite Zeile bezeichnet das zu versendende Dokument, und die dritte Zeile bezeichnet die das Dokument empfangende Aktivität. Die Aktivitätsnamen sind aus Platzgründen teilweise abgekürzt. Die Kosten pro Nutzung eines Kommunikationskanals (vorletzte Spalte) setzen sich aus der Summe der vergebenen Kostenpunkte (vgl. Abschnitt 4.1.2, Seite 157) zusammen. In der letzten mit NH bezeichneten Spalte ist die grob geschätzte Nutzungshäufigkeit des jeweiligen Kommunikationskanals pro Entwicklungsprojekt auf einer Skala von 1 bis 10 angegeben.

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Zur Abnahme freigeben Abnahmefreigabe Abnahme m. d. Kunden	1	1	0	0	0	1	3	2
Anf.-Analyse erstellen Anforderungsdokument Spezifikation erstellen	1	0	0	0	0	1	3	1
Anf.-Analyse erstellen Anforderungsdokument Vorh. Komp. auswählen	1	0	1	0	1	0	4	1
Anf.-Analyse erstellen Anforderungsdokument Versionsmanagement (AE)	1	0	0	0	1	0	4	1
Anf.-Analyse erstellen Anforderungsdokument Releaseplanung (AE)	1	0	0	0	1	0	3	1

Tabelle A.2: Kommunikationskanäle zwischen Kundenlokation und Anwendungsentwicklung

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Anf.-Analyse erstellen Anforderungsdokument Reviewdurchführung	1	0	0	1	1	1	3	5
Reviewdurchführung Review Anforderungsdok. Anf.-Analyse erstellen	1	0	1	0	1	0	4	5

Tabelle A.3: Kommunikationskanäle zwischen Kundenlokation und Qualitätsmanagement-Z

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Abnahme m. d. Kunden Abnahmebericht Projektabschluss	1	0	1	0	1	0	4	2
Kommunik. m. d. Kunden Erste Anforderungen Projektplanerstellung	0	0	0	1	0	1	2	3
Kommunik. m. d. Kunden Erste Anforderungen Teamzusammenstellung	0	1	0	1	0	1	2	3
Kommunik. m. d. Kunden Erste Anforderungen Ressourcenplanung	0	1	0	1	0	1	2	3
Anf.-Analyse erstellen Anfrage GP-Modell G.-Prozess modellieren	0	1	0	0	0	0	2	1
G.-Prozess modellieren Geschäftsprozessmodell Anf.-Analyse erstellen	1	0	1	0	0	0	3	1

Tabelle A.4: Kommunikationskanäle zwischen Kundenlokation und Managementlokation

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Richtlinienerstellung Richtlinien Richtl.erweit. vorschl.	1	0	0	0	1	0	3	2

Tabelle A.5: Kommunikationskanäle zwischen Qualitätsmanagement-A und Qualitätsmanagement-Z

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Releaseplanung (AE) Releaseplanung Projektplanerstellung	1	0	1	0	0	0	3	2
Buy/Build-Ent. treffen Buy/Build-Entscheidung Komponente kaufen	1	1	0	1	0	1	2	5
Projekt einschätzen Projekteinschätzung Releaseplanung (AE)	1	1	0	1	1	0	2	2
Komponente anfordern Komponentenanforderung Markt analysieren	1	0	1	0	0	1	4	2
Komponente anfordern Komponentenanforderung Entsch. neu/anpassen	1	1	0	0	0	0	2	5
Feedback an DE geben Feedback an DE Wiederverw.-repos. verw.	0	0	1	1	0	0	2	4
Feedback an DE geben Feedback an DE Wiederverw.-repos. verw.	0	0	1	1	0	0	2	4
Spezifikation erstellen Anfrage an Repository Wiederverw.-repos. verw.	0	1	0	1	0	1	2	1
Wiederverw.-repos. verw. Feedback v. Repository Spezifikation erstellen	1	0	1	0	0	0	3	2
Systemarchit. erstellen Anfrage an Repository Wiederverw.-repos. verw.	0	1	0	1	0	1	2	1

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Wiederverw.-repos. verw. Feedback v. Repository Systemarchit. erstellen	1	0	1	0	0	0	3	2
vorh. Komp. auswählen Anfrage an Repository Wiederverw.-repos. verw.	0	1	0	1	0	1	2	1
Spezifikation erstellen Spezifikationsdokument Wiederverw.-repos. verw.	1	0	1	0	0	0	3	1
Systemarchit. erstellen Systemarchitektur Wiederverw.-repos. verw.	1	0	1	0	0	0	3	1

Tabelle A.6: Kommunikationskanäle zwischen Anwendungsentwicklung und Managementlokation

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Testdurchführung Fehlerprot. install. SW-Syst. System installieren	1	0	1	0	0	1	4	3
Testdurchführung Fehlerprot. integr. SW-Syst. System integrieren	1	0	1	0	0	1	4	3
Reviewdurchführung Review Releaseplanung Releaseplanung (AE)	1	0	1	0	0	1	4	4
techn. Komp.-anf. erstellen techn. Komp.-anforderung Reviewdurchführung	1	0	1	0	1	0	4	2
System installieren installiertes Softwaresystem Testvorbereitung	1	0	0	0	0	0	2	2

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
System integrieren integriertes Softwaresyst. Testvorbereitung	0	0	0	0	0	0	2	2
Releaseplanung Releaseplanung Reviewdurchführung	1	0	0	0	0	1	3	2
Reviewdurchführung Review Spezifikationsdok. Spezifikation erstellen	0	0	0	0	1	1	4	3
Reviewdurchführung Review Systemarchitektur Systemarchitektur erstellen	0	0	0	0	1	1	4	3
Reviewdurchführung Review techn. Komp.-anf. Komponentenanf. erstellen	0	0	0	0	1	1	4	4
Spezifikation erstellen Spezifikationsdokument Reviewdurchführung	1	0	0	0	1	1	4	2
Systemarchit. erstellen Systemarchitektur Reviewdurchführung	1	0	0	0	1	1	4	2

Tabelle A.7: Kommunikationskanäle zwischen Anwendungsentwicklung und Qualitätsmanagement-Z

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Fehlerkorrektur getestete Komponente Komponente hinzufügen	1	0	1	0	0	0	3	1

Tabelle A.8: Kommunikationskanäle zwischen Anwendungsentwicklung und Komponentenentwicklung-1 bis -5

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Testdurchführung Fehlerprot. neue Komp. Fehlerkorrektur	1	0	0	0	0	1	3	4
Release erstellen (CE) Neue Komponente Testvorbereitung	1	0	1	0	0	1	4	2
Richtlinien anfragen Anfrage n. Richtlinien Richtlinienerstellung	0	1	0	1	1	1	3	1
Richtlinienerstellung Richtlinien Richtlinien anfragen	1	0	0	0	0	0	2	1

Tabelle A.9: Kommunikationskanäle zwischen Komponentenentwicklung-1, -3, -4, -5 und Qualitätsmanagement-Z

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Testdurchführung Fehlerprot. neue Komp. Fehlerkorrektur	1	0	0	0	0	1	3	4
Release erstellen (CE) Neue Komponente Testvorbereitung	1	0	1	0	0	1	4	2
Richtlinien anfragen Anfrage n. Richtlinien Richtlinienerstellung	0	1	0	1	1	1	3	1
Richtlinienerstellung Richtlinien Richtlinien anfragen	1	0	0	0	0	0	2	1

Tabelle A.10: Kommunikationskanäle zwischen Komponentenentwicklung-2 und Qualitätsmanagement-A

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Entsch. neu/anpassen Auftrag Komp.-anpass. Komponentenanpassung	1	0	1	0	1	0	4	1
Entsch. neu/anpassen Auftrag Komp.-anpass. Releaseplanung (CE)	1	0	1	0	1	0	4	1
Entsch. neu/anpassen Auftrag Komp.-entw. Clientkomp. Grobentw.	1	0	1	0	1	0	4	3
Entsch. neu/anpassen Auftrag Komp.-entw. Serverkomp. Grobentw.	1	0	1	0	1	0	4	1
Entsch. neu/anpassen Auftrag Komp.-entw. Releaseplanung (CE)	1	0	1	0	1	0	4	1
Versionsmanagement versionierte Komponente Wiederverw.-repos. verw.	1	0	1	0	0	0	3	1

Tabelle A.11: Kommunikationskanäle zwischen Komponentenentwicklung-1, -2, -3, -4, -5 und Managementlokation

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Richtl.-erweit. vorschl. Vorschlag Richtlinienerw. R.-vorschlag weiterleiten	0	0	0	1	0	0	1	2

Tabelle A.12: Kommunikationskanäle zwischen Qualitätsmanagement-A und Managementlokation

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Risikomanagement Analysen und Berichte Projektst. und -kontrolle	1	0	1	0	1	0	4	5
Projektplanerstellung Anfrage an DE Projekt einschätzen	0	1	0	1	0	1	2	1
G.-Prozess modellieren Anfrage an Repository Wiederverw.-rep. verw.	0	1	0	1	0	1	2	1
Wiederverw.-rep. verw. Feedback v. Repository G.-Prozess modellieren	0	0	1	0	1	1	5	2
Projekt einschätzen Anfrage an Repository Wiederverw.-rep. verw.	0	1	0	1	0	1	2	1
Wiederverw.rep. verw. Feedback v. Repository Projekt einschätzen	0	0	1	0	1	0	4	2
Ressourcenplanung Ausstattung Projekt initialisieren	1	0	0	0	0	1	3	1
Buy/Build-Entsch. treffen Buy/Build-Entscheidung CE m. Entw. beauftr.	1	1	0	0	0	0	2	5
CE m. Entw. beauftr. Entwicklungsbeauftragung Entsch. neu/anpassen	1	0	1	0	1	0	4	5
G.-Prozess modellieren Geschäftsprozessmodell Wiederverw.-rep. verw.	1	0	1	0	0	0	3	1

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Markt analysieren Marktanalyse Buy/Build-Entsch. treffen	1	0	0	0	1	1	4	1
Projekt einschätzen Projekteinschätzung Projektplanerstellung	1	0	0	1	1	0	2	2
Projekt einschätzen Projekteinschätzung Risikomanagement	1	0	0	0	1	0	3	2
Projekt einschätzen Projekteinschätzung Markt analysieren	1	1	0	0	1	1	4	2
Projekt initialisieren Projektinitialisierung Projektst. u. -kontrolle	0	1	0	0	0	1	3	1
Projektst. u. -kontrolle Projektplan Projekt initialisieren	1	1	0	0	0	0	2	2
Projektplanerstellung Projektplan Projektst. u. -kontrolle	1	0	0	0	0	1	3	2
Projektst. u. -kontrolle Projektstatus Projektdarst. n. aussen	1	0	0	1	0	0	1	5
Teamzusammenstellung Projektteam Projekt initialisieren	1	0	0	0	1	1	4	1

Tabelle A.13: Kommunikationskanäle am Standort *Managementlokation*

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Kommunik. m. d. Kunden Erste Anforderungen Anf.-analyse erstellen	0	1	0	1	0	1	2	3

Tabelle A.14: Kommunikationskanäle am Standort *Kundenlokation*

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Reviewauswertung Reviewauswertung Richtlinienerstellung	1	0	1	0	1	0	4	10
Richtlinienerstellung Richtlinien Testauswertung	1	0	0	0	0	0	2	1
Richtlinienerstellung Richtlinien Reviewauswertung	1	0	0	0	0	0	2	1
Testauswertung Testauswertung Richtlinienerstellung	1	0	1	0	1	0	4	6
Testplanerstellung Testplan Testablaufverfolgung	1	0	0	0	1	0	3	1
Testplanerstellung Testplan Reviewauswertung	1	0	0	0	1	0	3	1
Testplanerstellung Testplan Testauswertung	1	0	0	0	1	0	3	1
Reviewdurchführung Rev. Anforderungsdok. Reviewauswertung	1	0	1	0	0	0	3	5

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Reviewdurchführung Rev. Spezifikationsdok. Reviewauswertung	1	0	1	0	0	0	3	3
Reviewdurchführung Review Systemarchitektur Reviewauswertung	1	0	1	0	0	0	3	3
Reviewdurchführung Review Releaseplanung Reviewauswertung	1	0	1	0	0	0	3	2
Reviewdurchführung Rev. techn. Komp.-anf. Reviewauswertung	1	0	1	0	0	0	3	4
Testdurchführung Fehlerprot. neue Komp. Testauswertung	1	0	1	0	0	0	3	4
Testdurchführung Fehlerprot. install. SW-Syst. Testauswertung	1	0	1	0	0	0	3	3
Testdurchführung Fehlerprot. integr. SW-Syst. Testauswertung	1	0	1	0	0	0	3	3
Richtlinienerstellung Richtlinien Reviewdurchführung	1	0	0	0	1	1	4	1
Richtlinienerstellung Richtlinien Testdurchführung	1	0	0	0	1	1	4	1
Richtlinienerstellung Richtlinien Testvorbereitung	1	0	0	0	1	1	4	1

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Reviewdurchführung Status Review Testablaufverfolgung	1	1	0	1	0	1	2	3
Reviewdurchführung Status Test Testablaufverfolgung	1	1	0	1	0	1	2	3
Testvorbereitung Testbeschreibung Testdurchführung	1	0	0	0	1	1	4	1
Testplanerstellung Testplan Testvorbereitung	1	0	1	0	0	1	4	1
Testvorbereitung Testbeschreibung Testauswertung	1	0	0	0	0	1	3	1

Tabelle A.15: Kommunikationskanäle am Standort *Qualitätsmanagement-Z*

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Komp.-anpassung Komp.-implementation Release erstellen	1	0	0	0	1	0	3	1
Klassenintegration Komp.-implementation Release erstellen	1	0	0	0	1	0	3	1
Serverkomp. Grobentw. Komp.-integrationsplan Feinentwurf	1	0	0	0	1	1	4	2
Serverkomp. Grobentw. Komp.-integrationsplan DB-Entw. u. D.-migration	1	0	0	0	1	1	4	2

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Clientkomp. Grobentw. Komp.-integrationsplan Feinentwurf	1	0	0	0	1	1	4	2
Clientkomp. Grobentw. Komp.-integrationsplan DB-Entw. u. D.-migrat.	1	0	0	0	1	1	4	2
Releaseplanung (CE) Komp.-releaseplanung Release erstellen	1	0	0	0	1	1	4	2
Implementierung Quellcode Klassenintegration	1	0	1	1	1	1	4	1
Serverkomp. Grobentw. Schnittstellenbeschreibung Feinentwurf	1	0	1	0	1	1	5	2
Clientkomp. Grobentw. Schnittstellenbeschreibung Feinentwurf	1	0	1	0	1	1	5	2
Clientkomp. Grobentw. SW-Archit. Clientkomp. Feinentwurf	1	0	0	1	1	1	3	2
Serverkomp. Grobentw. SW-Archit. Serverkomp. Feinentwurf	1	0	0	1	1	1	3	2
Serverkomp. Grobentwurf SW-Archit. Serverkomp. DB-Entw. u. D.-migration	1	0	0	1	1	1	3	2
Feinentwurf SW-Entw. Clientkomp. Implementierung	1	0	0	0	1	1	4	2

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Feinentwurf SW-Entw. Serverkomp. Implementierung	1	0	0	0	1	1	4	2
DB-Entw. u. D.-migration SW-Entw. u. D.-migration DB-Implementierung	1	0	0	0	1	1	4	2
Fehlerkorrektur getestete Komponente Versionsmanagement (CE)	1	0	0	0	1	1	4	3
Release erstellen neue Komponente Fehlerkorrektur	1	0	0	0	1	1	4	3

Tabelle A.16: Kommunikationskanäle an den Standorten *Komponenten-entwicklung-1* bis *-5*

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Komponente kaufen gekaufte Komponente Komponente hinzufügen	0	0	0	0	1	0	3	3
System installieren install. SW-System Zur Abnahme freigeben	1	1	0	0	0	0	2	1
System installieren install. SW-System Release erstellen	1	0	0	0	1	0	3	1
System integrieren integr. SW-System System installieren	0	1	0	1	1	1	3	1
System integrieren integr. SW-System Feedback an DE geben	0	0	0	0	1	0	3	1

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
System integrieren integr. SW-System Versionsmanagement (AE)	0	0	0	0	1	0	3	1
Release erstellen (AE) Konfiguration Versionsmanagement (AE)	1	1	0	0	1	0	3	1
Releaseplanung (AE) Releaseplanung Release erstellen	1	0	0	0	0	1	3	2
techn. Komp.-anf. erst. techn. Komp.-anford. Komponente anfordern	1	0	0	0	1	0	3	1
Spezifikation erstellen Spezifikationsdokument Systemarchit. erstellen	1	0	0	0	1	1	4	1
Spezifikation erstellen Spezifikationsdokument Feedback an DE geben	1	0	0	0	1	0	3	1
Spezifikation erstellen Spezifikationsdokument Releaseplanung (AE)	1	0	0	0	1	0	3	1
Spezifikation erstellen Spezifikationsdokument Versionsmanagement (AE)	1	0	0	0	1	0	3	1
Systemarchit. erstellen Systemarchitektur techn. Komp.-anf. erst.	1	0	0	0	1	1	4	1
Systemarchit. erstellen Systemarchitektur Komponenten auswählen	1	0	0	0	1	0	3	1

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Systemarchit. erstellen Systemarchitektur Versionsmanagement (AE)	1	0	0	0	1	0	3	1
Systemarchit. erstellen Systemarchitektur Feedback an DE geben	1	0	0	0	1	0	3	1
Komponente hinzufügen projektbez. Komp.-menge Feedback an DE geben	1	0	0	0	1	0	3	1
Komponente hinzufügen projektbez. Komp.-menge vorh. Komp. auswählen	1	0	0	0	1	1	4	2
Komponente hinzufügen projektbez. Komp.-menge System integrieren	1	0	0	0	1	1	4	1

Tabelle A.17: Kommunikationskanäle am Standort *Anwendungsentwicklung*

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Reviewauswertung Reviewauswertung Richtl.-erweit. vorschl.	1	0	1	0	1	0	4	4
Testauswertung Testauswertung Richtl.-erweit. vorschl.	1	0	1	0	1	0	4	4
Testplanerstellung Testplan Testablaufverfolgung	1	0	0	0	1	0	3	1
Testplanerstellung Testplan Reviewauswertung	1	0	0	0	1	0	3	1

Kommunikationskanal	**per**	**synch**	**coded**	**change**	**priv**	**mult**	**Kosten**	**NH**
Testplanerstellung Testplan Testauswertung	1	0	0	0	1	0	3	1
Reviewdurchführung Rev. Komp.releaseplan. Reviewauswertung	1	0	0	0	1	0	3	2
Testdurchführung Fehlerprot. neue Komp. Testauswertung	1	0	1	0	0	0	3	3
Reviewdurchführung Status Review Testablaufverfolgung	1	1	0	1	0	1	2	3
Reviewdurchführung Status Test Testablaufverfolgung	1	1	0	1	0	1	2	3
Testvorbereitung Testbeschreibung Testdurchführung	1	0	0	0	1	1	4	1
Testplanerstellung Testplan Testvorbereitung	1	0	1	0	0	1	4	1
Testvorbereitung Testbeschreibung Testauswertung	1	0	0	0	0	1	3	1
Richtl.-erweit. vorschl. Richtlinien Testauswertung	1	0	0	0	0	0	2	1
Richtl.-erweit. vorschl. Richtlinien Reviewauswertung	1	0	0	0	0	0	2	1

Kommunikationskanal	per	synch	coded	change	priv	mult	Kosten	NH
Richtl.-erweit. vorschl. Richtlinien Reviewdurchführung	1	0	0	0	1	1	4	1
Richtl.-erweit. vorschl. Richtlinien Testdurchführung	1	0	0	0	1	1	4	1
Richtl.-erweit. vorschl. Richtlinien Testvorbereitung	1	0	0	0	1	1	4	1

Tabelle A.18: Kommunikationskanäle am Standort *Qualitätsmanagement-A*

Die Tabellen A.19 bis A.70 zeigen verschiedene Detailberechnungen, die zur Messung der Kkf verschiedener Optimierungsansätze erforderlich sind.

Aufbau eines neuen Standortes: (Tabellen A.19 bis A.30)

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation	4	3
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung	Managementlokation	4	4
Anwendungsentwicklung	Qualitätsmanagement-Z	12	5
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	5	1
Managementlokation	Komponentenentwicklung-2	5	1
Managementlokation	Komponentenentwicklung-3	5	1
Managementlokation	Komponentenentwicklung-4	5	1
Managementlokation	Komponentenentwicklung-5	5	1
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
neuer Standort	Kundenlokation	2	2
neuer Standort	Anwendungsentwicklung	9	5

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
neuer Standort	Managementlokation	4	4
neuer Standort	Komponentenentwicklung-1	1	1
neuer Standort	Komponentenentwicklung-2	1	1
neuer Standort	Komponentenentwicklung-3	1	1
neuer Standort	Komponentenentwicklung-4	1	1
neuer Standort	Komponentenentwicklung-5	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.19: Kopplung zwischen zwei Orten – Alternative1

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	13
Managementlokation	42	9
neuer Standort	20	5
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.20: Kopplung eines Ortes – Alternative1

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation	10	7
neuer Standort	5	4
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5

Ort	Kohäsionsdichte	Kohäsionskomplexität
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.21: Kohäsion eines Ortes – Alternative1

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation	7,72
neuer Standort	23,55
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.22: Kommunikationskostenfaktoren aller Orte – Alternative1

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	3	3
Kundenlokation	Managementlokation	6	5
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung	Managementlokation	11	6
Anwendungsentwicklung	Qualitätsmanagement-Z	10	4
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
neuer Standort	Kundenlokation	2	1
neuer Standort	Anwendungsentwicklung	5	1
neuer Standort	Managementlokation	2	1
neuer Standort	Qualitätsmanagement-Z	2	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.23: Kopplung zwischen zwei Orten – Alternative2

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	34	11
Managementlokation	54	9
neuer Standort	11	5
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.24: Kopplung eines Ortes– Alternative2

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
Kundenlokation	1	1
Anwendungsentwicklung	18	4
Managementlokation	19	12
neuer Standort	2	2
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5

Ort	Kohäsionsdichte	Kohäsionskomplexität
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.25: Kohäsion eines Ortes – Alternative2

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung	10,94
Managementlokation	6,14
neuer Standort	18,7
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.26: Kommunikationskostenfaktoren aller Orte – Alternative2

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation	6	5
Anwendungsentwicklung	Managementlokation	13	7
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
neuer Standort	Kundenlokation	2	2
neuer Standort	Anwendungsentwicklung	12	5
neuer Standort	Qualitätsmanagement-Z	12	6
neuer Standort	Komponentenentwicklung-1	2	2
Komponentenentwicklung-1	Qualitätsmanagement-Z	2	2
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	2	2
Komponentenentwicklung-4	Qualitätsmanagement-Z	2	2
Komponentenentwicklung-5	Qualitätsmanagement-Z	2	2
Komponentenentwicklung-2	neuer Standort	2	2
Komponentenentwicklung-3	neuer Standort	2	2
Komponentenentwicklung-4	neuer Standort	2	2
Komponentenentwicklung-5	neuer Standort	2	2
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.27: Kopplung zwischen zwei Orten – Alternative3

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	13
Managementlokation	54	9
neuer Standort	36	9
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	25	8
Qualitätsmanagement-A	6	6

Tabelle A.28: Kopplung eines Ortes – Alternative3

Ort	Kohäsionsdichte	Kohäsionskomplexität
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation	19	12
neuer Standort	10	3
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	1	1
Qualitätsmanagement-A	17	7

Tabelle A.29: Kohäsion eines Ortes – Alternative3

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation	6,14
neuer Standort	1209,6
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	1241,66
Qualitätsmanagement-A	0,15

Tabelle A.30: Kommunikationskostenfaktoren aller Orte – Alternative3

Zusammenlegen mehrerer Orte: (Tabellen A.31 bis A.54)

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung inkl. Management	11	8
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung inkl. Management	Qualitätsmanagement-Z	16	6

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Anwendungsentwicklung inkl. Management	Komponentenentwicklung-1	7	2
Anwendungsentwicklung inkl. Management	Komponentenentwicklung-2	7	2
Anwendungsentwicklung inkl. Management	Komponentenentwicklung-3	7	2
Anwendungsentwicklung inkl. Management	Komponentenentwicklung-4	7	2
Anwendungsentwicklung inkl. Management	Komponentenentwicklung-5	7	2
Anwendungsentwicklung inkl. Management	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.31: Kopplung zwischen zwei Orten – Fusionsalternative1

Ort	Kopplungsdichte	Kopplungskomplexität
Kundenlokation	13	9
Anwendungsentwicklung inkl. Management	62	12
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.32: Kopplung eines Ortes – Fusionsalternative1

Ort	Kohäsionsdichte	Kohäsionskomplexität
Kundenlokation	1	1
Anwendungsentwicklung inkl. Management	52	16
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.33: Kohäsion eines Ortes – Fusionsalternative1

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung inkl. Management	1,65
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.34: Kommunikationskostenfaktoren aller Orte – Fusionsalternative1

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung inkl. Komponentenentwicklung-1	5	4
Kundenlokation	Managementlokation	6	5
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Managementlokation	19	8
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Qualitätsmanagement-Z	16	6

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Komponentenentwicklung-2	1	1
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Komponentenentwicklung-3	1	1
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Komponentenentwicklung-4	1	1
Anwendungsentwicklung inkl. Komponentenentwicklung-1	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.35: Kopplung zwischen zwei Orten – Fusionsalternative2

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung inkl. Komponentenentwicklung-1	46	13
Managementlokation	54	9
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.36: Kopplung eines Ortes – Fusionsalternative2

Ort	Kohäsionsdichte	Kohäsionskomplexität
Kundenlokation	1	1
Anwendungsentwicklung inkl. Komponentenentwicklung-1	38	10
Managementlokation	19	12
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.37: Kohäsion eines Ortes – Fusionsalternative2

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung inkl. Komponentenentwicklung-1	1,51
Managementlokation	6,14
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.38: Kommunikationskostenfaktoren aller Orte – Fusionsalternative2

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung inkl. Qualitätsmanagement-Z	7	5
Kundenlokation	Managementlokation	6	5
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Managementlokation	17	10
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Komponentenentwicklung-1	5	5

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Komponentenentwicklung-2	1	1
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Komponentenentwicklung-3	5	5
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Komponentenentwicklung-4	5	5
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	Komponentenentwicklung-5	5	5
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-A	1	1
Qualitätsmanagement-A	Anwendungsentwicklung inkl. Qualitätsmanagement-Z	1	1

Tabelle A.39: Kopplung zwischen zwei Orten – Fusionsalternative3

Ort	Kopplungsdichte	Kopplungskomplexität
Kundenlokation	13	9
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	46	13
Managementlokation	54	9
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-A	6	6

Tabelle A.40: Kopplung eines Ortes – Fusionsalternative3

Ort	Kohäsionsdichte	Kohäsionskomplexität
Kundenlokation	1	1
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	55	11
Managementlokation	19	12

Ort	Kohäsionsdichte	Kohäsionskomplexität
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-A	17	7

Tabelle A.41: Kohäsion eines Ortes – Fusionsalternative3

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung inkl. Qualitätsmanagement-Z	0,57
Managementlokation	6,14
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-A	0,15

Tabelle A.42: Kommunikationskostenfaktoren aller Orte – Fusionsalternative3

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation inkl. Komponentenentwicklung-1	6	5
Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung	Managementlokation inkl. Komponentenentwicklung-1	14	7
Anwendungsentwicklung	Qualitätsmanagement-Z	12	5
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation inkl. Komponentenentwicklung-1	Komponentenentwicklung-2	6	2

Ort 1	**Ort 2**	**Kopplungs-**	**Kopplungs-**
Managementlokation inkl. Komponentenentwicklung-1	Komponentenentwicklung-3	6	2
Managementlokation inkl. Komponentenentwicklung-1	Komponentenentwicklung-4	6	2
Managementlokation inkl. Komponentenentwicklung-1	Komponentenentwicklung-5	6	2
Managementlokation inkl. Komponentenentwicklung-1	Qualitätsmanagement-Z	8	5
Managementlokation inkl. Komponentenentwicklung-1	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.43: Kopplung zwischen zwei Orten – Fusionsalternative4

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	13
Managementlokation inkl. Komponentenentwicklung-1	53	13
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.44: Kopplung eines Ortes – Fusionsalternative4

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation inkl. Komponentenentwicklung-1	43	14
Komponentenentwicklung-2	18	5

Ort	Kohäsionsdichte	Kohäsionskomplexität
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.45: Kohäsion eines Ortes – Fusionsalternative4

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation inkl. Komponentenentwicklung-1	1,12
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.46: Kommunikationskostenfaktoren aller Orte – Fusionsalternative4

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation inkl. Qualitätsmanagement-Z	8	7
Anwendungsentwicklung	Managementlokation inkl. Qualitätsmanagement-Z	25	11
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation inkl. Qualitätsmanagement-Z	Komponentenentwicklung-1	10	6
Managementlokation inkl. Qualitätsmanagement-Z	Komponentenentwicklung-2	6	2

Ort 1	**Ort 2**	**Kopplungs-**	**Kopplungs-**
Managementlokation inkl. Qualitätsmanagement-Z	Komponentenentwicklung-3	10	6
Managementlokation inkl. Qualitätsmanagement-Z	Komponentenentwicklung-4	10	6
Managementlokation inkl. Qualitätsmanagement-Z	Komponentenentwicklung-5	10	6
Managementlokation inkl. Qualitätsmanagement-Z	Qualitätsmanagement-A	1	1
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4

Tabelle A.47: Kopplung zwischen zwei Orten – Fusionsalternative5

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	13
Managementlokation inkl. Qualitätsmanagement-Z	74	15
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-A	6	6

Tabelle A.48: Kopplung eines Ortes – Fusionsalternative5

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation inkl. Qualitätsmanagement-Z	46	15
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-A	17	7

Tabelle A.49: Kohäsion eines Ortes – Fusionsalternative5

Ort	Kkf
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation inkl. Qualitätsmanagement-Z	3,61
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-A	0,15

Tabelle A.50: Kommunikationskostenfaktoren aller Orte – Fusionsalternative5

Ort 1	Ort 2	Kopplungs-dichte	Kopplungs-komplexität
Kundenlokation	Anwendungsentwicklung	5	4
Kundenlokation	Managementlokation	6	5
Kundenlokation	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	2	2
Anwendungsentwicklung	Managementlokation	13	7
Anwendungsentwicklung	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	13	6
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	10	5
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	4	4

Ort 1	**Ort 2**	**Kopplungs-**	**Kopplungs-**
Komponentenentwicklung-5	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	1	1

Tabelle A.51: Kopplung zwischen zwei Orten – Fusionsalternative6

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
Kundenlokation	13	9
Anwendungsentwicklung	35	13
Managementlokation	54	9
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	38	9
Qualitätsmanagement-A	6	6

Tabelle A.52: Kopplung eines Ortes – Fusionsalternative6

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation	19	12
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	45	12
Qualitätsmanagement-A	17	7

Tabelle A.53: Kohäsion eines Ortes – Fusionsalternative6

Ort	**Kkf**
Kundenlokation	186,33
Anwendungsentwicklung	8,87
Managementlokation	6,14
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z inkl. Komponentenentwicklung-1	0,77
Qualitätsmanagement-A	0,15

Tabelle A.54: Kommunikationskostenfaktoren aller Orte – Fusionsalternative6

Verlagerung einer Aktivität: (Tabellen A.55 bis A.70)

Die in den nachfolgenden Tabellen mit einem Sternchen versehenen Standorte beinhalten die verlagerte Aktivität *Anforderungsanalyse*.

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
verkleinerte Kundenlokation	Anwendungsentwicklung*	2	2
verkleinerte Kundenlokation	Managementlokation	4	3
Anwendungsentwicklung*	Managementlokation	15	7
Anwendungsentwicklung*	Qualitätsmanagement-Z	14	6
Anwendungsentwicklung*	Komponentenentwicklung-1	1	1
Anwendungsentwicklung*	Komponentenentwicklung-2	1	1
Anwendungsentwicklung*	Komponentenentwicklung-3	1	1
Anwendungsentwicklung*	Komponentenentwicklung-4	1	1
Anwendungsentwicklung*	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.55: Kopplung zwischen zwei Orten – Verlagerungsalternative1

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
verkleinerte Kundenlokation	6	4
Anwendungsentwicklung*	36	12
Managementlokation	54	9
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.56: Kopplung eines Ortes – Verlagerungsalternative1

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
verkleinerte Kundenlokation	0	0
Anwendungsentwicklung*	24	7
Managementlokation	19	12
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.57: Kohäsion eines Ortes – Verlagerungsalternative1

Ort	**Kkf**
verkleinerte Kundenlokation	32
Anwendungsentwicklung*	4,05
Managementlokation	6,14
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.58: Kommunikationskostenfaktoren aller Orte – Verlagerungsalternative1

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
verkleinerte Kundenlokation	Anwendungsentwicklung	1	1
verkleinerte Kundenlokation	Managementlokation*	5	3
Anwendungsentwicklung	Managementlokation*	17	10
Anwendungsentwicklung	Qualitätsmanagement-Z	12	5
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation*	Komponentenentwicklung-1	6	2
Managementlokation*	Komponentenentwicklung-2	6	2
Managementlokation*	Komponentenentwicklung-3	6	2
Managementlokation*	Komponentenentwicklung-4	6	2
Managementlokation*	Komponentenentwicklung-5	6	2
Managementlokation*	Qualitätsmanagement-Z	6	4
Managementlokation*	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1

Tabelle A.59: Kopplung zwischen zwei Orten – Verlagerungsalternative2

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
verkleinerte Kundenlokation	6	4
Anwendungsentwicklung	35	13
Managementlokation*	59	13
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.60: Kopplung eines Ortes – Verlagerungsalternative2

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
verkleinerte Kundenlokation	0	0
Anwendungsentwicklung	20	6
Managementlokation*	21	12
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.61: Kohäsion eines Ortes – Verlagerungsalternative2

Ort	**Kkf**
verkleinerte Kundenlokation	32
Anwendungsentwicklung	8,87
Managementlokation*	6,45
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.62: Kommunikationskostenfaktoren aller Orte – Verlagerungsalternative2

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
verkleinerte Kundenlokation	Anwendungsentwicklung	1	1
verkleinerte Kundenlokation	Managementlokation	4	3
verkleinerte Kundenlokation	Qualitätsmanagement-Z*	1	1
Anwendungsentwicklung	Managementlokation	13	7
Anwendungsentwicklung	Qualitätsmanagement-Z*	16	6
Anwendungsentwicklung	Komponentenentwicklung-1	1	1
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1	6	2
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z*	6	5
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1	Qualitätsmanagement-Z*	4	4
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z*	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z*	4	4

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Komponentenentwicklung-5	Qualitätsmanagement-Z*	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z*	1	1

Tabelle A.63: Kopplung zwischen zwei Orten – Verlagerungsalternative3

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
verkleinerte Kundenlokation	6	4
Anwendungsentwicklung	35	13
Managementlokation	54	9
Komponentenentwicklung-1	11	6
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z*	40	11
Qualitätsmanagement-A	6	6

Tabelle A.64: Kopplung eines Ortes – Verlagerungsalternative3

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
verkleinerte Kundenlokation	1	1
Anwendungsentwicklung	20	6
Managementlokation	19	12
Komponentenentwicklung-1	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z*	25	9
Qualitätsmanagement-A	17	7

Tabelle A.65: Kohäsion eines Ortes – Verlagerungsalternative3

Ort	**Kkf**
verkleinerte Kundenlokation	32
Anwendungsentwicklung	8,87
Managementlokation	6,14
Komponentenentwicklung-1	0,26
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z*	1,98
Qualitätsmanagement-A	0,15

Tabelle A.66: Kommunikationskostenfaktoren aller Orte – Verlagerungsalternative3

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
verkleinerte Kundenlokation	Anwendungsentwicklung	1	1
verkleinerte Kundenlokation	Managementlokation	4	3
verkleinerte Kundenlokation	Qualitätsmanagement-Z	2	2
Anwendungsentwicklung	Managementlokation	13	7
Anwendungsentwicklung	Qualitätsmanagement-Z	12	5
Anwendungsentwicklung	Komponentenentwicklung-1*	5	5
Anwendungsentwicklung	Komponentenentwicklung-2	1	1
Anwendungsentwicklung	Komponentenentwicklung-3	1	1
Anwendungsentwicklung	Komponentenentwicklung-4	1	1
Anwendungsentwicklung	Komponentenentwicklung-5	1	1
Managementlokation	Komponentenentwicklung-1*	8	3
Managementlokation	Komponentenentwicklung-2	6	2
Managementlokation	Komponentenentwicklung-3	6	2
Managementlokation	Komponentenentwicklung-4	6	2
Managementlokation	Komponentenentwicklung-5	6	2
Managementlokation	Qualitätsmanagement-Z	4	3
Managementlokation	Qualitätsmanagement-A	1	1
Komponentenentwicklung-1*	Qualitätsmanagement-Z	6	6
Komponentenentwicklung-2	Qualitätsmanagement-A	4	4
Komponentenentwicklung-3	Qualitätsmanagement-Z	4	4
Komponentenentwicklung-4	Qualitätsmanagement-Z	4	4

Ort 1	**Ort 2**	**Kopplungs-dichte**	**Kopplungs-komplexität**
Komponentenentwicklung-5	Qualitätsmanagement-Z	4	4
Qualitätsmanagement-A	Qualitätsmanagement-Z	1	1
verkleinerte Kundenlokation	Komponentenentwicklung-1*	1	1

Tabelle A.67: Kopplung zwischen zwei Orten – Verlagerungsalternative4

Ort	**Kopplungsdichte**	**Kopplungskomplexität**
verkleinerte Kundenlokation	7	4
Anwendungsentwicklung	35	13
Managementlokation	54	9
Komponentenentwicklung-1*	20	11
Komponentenentwicklung-2	11	6
Komponentenentwicklung-3	11	6
Komponentenentwicklung-4	11	6
Komponentenentwicklung-5	11	6
Qualitätsmanagement-Z	35	8
Qualitätsmanagement-A	6	6

Tabelle A.68: Kopplung eines Ortes – Verlagerungsalternative4

Ort	**Kohäsionsdichte**	**Kohäsionskomplexität**
verkleinerte Kundenlokation	0	0
Anwendungsentwicklung	20	6
Managementlokation	19	12
Komponentenentwicklung-1*	18	5
Komponentenentwicklung-2	18	5
Komponentenentwicklung-3	18	5
Komponentenentwicklung-4	18	5
Komponentenentwicklung-5	18	5
Qualitätsmanagement-Z	23	8
Qualitätsmanagement-A	17	7

Tabelle A.69: Kohäsion eines Ortes – Verlagerungsalternative4

Ort	Kkf
verkleinerte Kundenlokation	32
Anwendungsentwicklung	8,87
Managementlokation	6,14
Komponentenentwicklung-1*	0,97
Komponentenentwicklung-2	0,26
Komponentenentwicklung-3	0,26
Komponentenentwicklung-4	0,26
Komponentenentwicklung-5	0,26
Qualitätsmanagement-Z	2,28
Qualitätsmanagement-A	0,15

Tabelle A.70: Kommunikationskostenfaktoren aller Orte – Verlagerungsalternative4

A.3 Detailinformationen zur dynamischen Analyse

In diesem Abschnitt sind die zur Parametrisierung des Simulationsmodells verwendeten Daten detailliert aufgeführt. Dazu ist für jedes separat modellierte Petrinetz eine entsprechende Abbildung eingefügt, die mit Hilfe der Tabellen A.71 bis A.76 Auskunft über Zeitbedarfe der modellierten Aktivitäten und Volumina der ausgetauschten Informationen geben. Den Zeitbedarfen sowie den Volumina liegen Schätzungen für kleine (ca. 100 Personentage), mittlere (ca. 500 Personentage) und große Projekte (ca. 1000 Personentage) zugrunde. Nähere Begründungen für bestimmte Volumina und Zeitbedarfe finden sich bei [Stö01b].

Ist in den Tabellen A.71 bis A.76 anstelle des Zeitbedarfs ein Volumen angegeben, so bedeutet dies, dass durch die Aktivität eine Kommunikation angestoßen wird, bei der eine Nachricht der angegebenen Größe übermittelt wird. Die in den Tabellen mit x.c() abstrakt angegebene Variable repräsentiert den Komplexitätswert, der während der Simulation zufällig und gleichverteilt auf einer Skala von 1 bis 10 zugewiesen wird.

Nach der Darstellung der Simulationsmodelle sowie zugehöriger Zeitbedarfe und Volumina sind abschließend zwei Kostenmodelle erläutert, die die Kosten der modellierten Kommunikationssysteme beeinflussen.

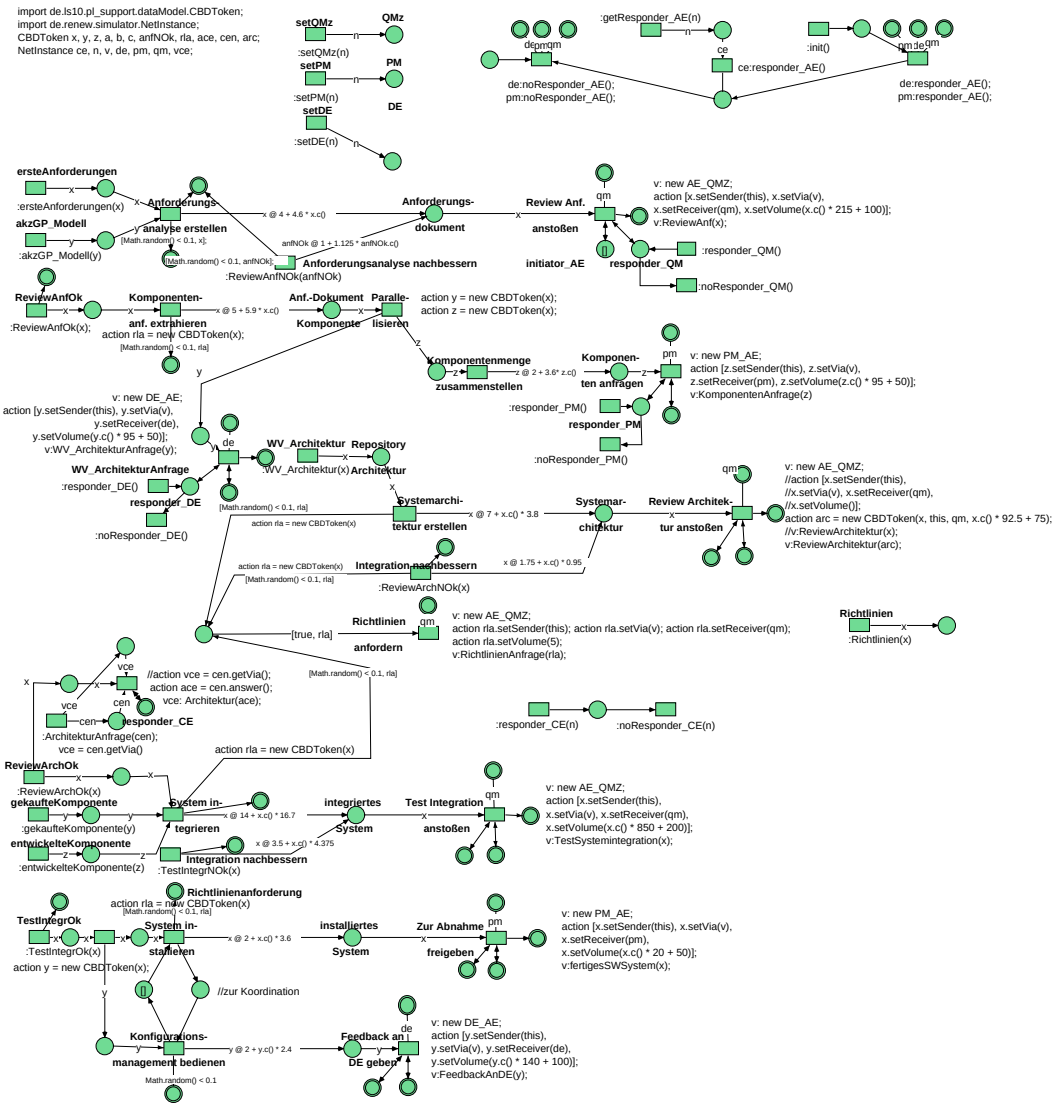


Abbildung A.1: Standort *Anwendungsentwicklung* im Simulationsmodell

Aktivität	**Zeitbedarf**	**Volumen**
Anforderungsanalyse erstellen	4 - 45 Tage $\hat{=}$ 4 + x.c() $\times$ 4.1	
Review Anforderungs-dokument anstoßen		75 - 2.250 kB $\hat{=}$ 100 + x.c() $\times$ 215
Anforderungsanalyse nachbessern	1 - 11 Tage $\hat{=}$ 1 + x.c() $\times$ 1.0	
Komponentenan-forderungen extrahieren	5 - 64 Tage $\hat{=}$ 5 + x.c() $\times$ 5.9	
Zusammenstellung der Komponentenmenge	2 - 38 Tage $\hat{=}$ 2 + x.c() $\times$ 3.6	
Komponente anfordern		50 - 100 kB $\hat{=}$ 50 + x.c() $\times$ 95
Systemarchitektur erstellen	7 - 45 Tage $\hat{=}$ 7 + x.c() $\times$ 3.8	
Systemarchitektur nachbessern	2 - 11 Tage $\hat{=}$ 1.75 + x.c() $\times$ 0.95	
Review Architektur anstoßen		75 - 1000 kB $\hat{=}$ 75 + x.c() $\times$ 92.5
System integrieren	14 - 181 Tage $\hat{=}$ 14 + x.c() $\times$ 16.7	
Systemintegration nachbessern	4 - 45 Tage $\hat{=}$ 4 + x.c() $\times$ 4.1	
Integrationstest anstoßen		500 - 9000 kB $\hat{=}$ 500 + x.c() $\times$ 850
System installieren	2 - 38 Tage $\hat{=}$ 2 + x.c() $\times$ 3.6	
Richtlinien anfragen		5 kB $\hat{=}$ 5
Konfigurations-management	2 - 26 Tage $\hat{=}$ 2 + x.c() $\times$ 2.4	
Feedback an DE geben		100 - 1500 kB $\hat{=}$ 10 + x.c() $\times$ 140
Zur Abnahme freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7

Tabelle A.71: Parametrisierung des Standortes *Anwendungsentwicklung*

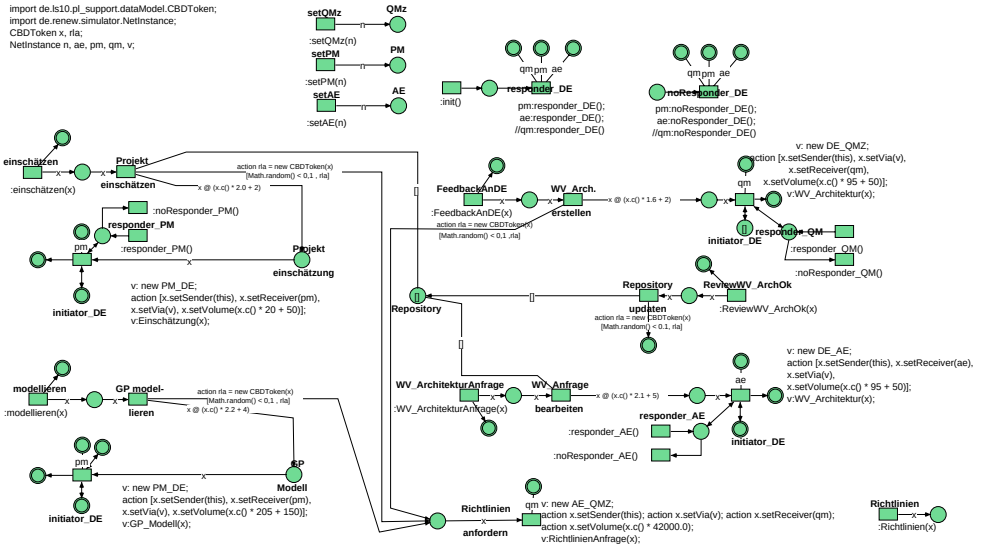


Abbildung A.2: Standort *Domain Engineering* im Simulationsmodell

Aktivität	**Zeitbedarf**	**Volumen**
Projekt einschätzen	2 - 22 Tage $\hat{=}$ 2 + x.c() $\times$ 2	
Geschäftsprozess modellieren	4 - 26 Tage $\hat{=}$ 4 + x.c() $\times$ 2.2	
Wiederverwendungs-architektur erstellen	1 - 7 Tage $\hat{=}$ 1 + x.c() $\times$ 0.7	
Wiederverwendungs-repository verwalten	0 Tage $\hat{=}$ 0	
Repository-Anfrage bearbeiten	5 - 26 Tage $\hat{=}$ 5 + x.c() $\times$ 21	
Richtlinien anfragen		5 kB $\hat{=}$ 5

Tabelle A.72: Parametrisierung des Standortes *Domain Engineering*

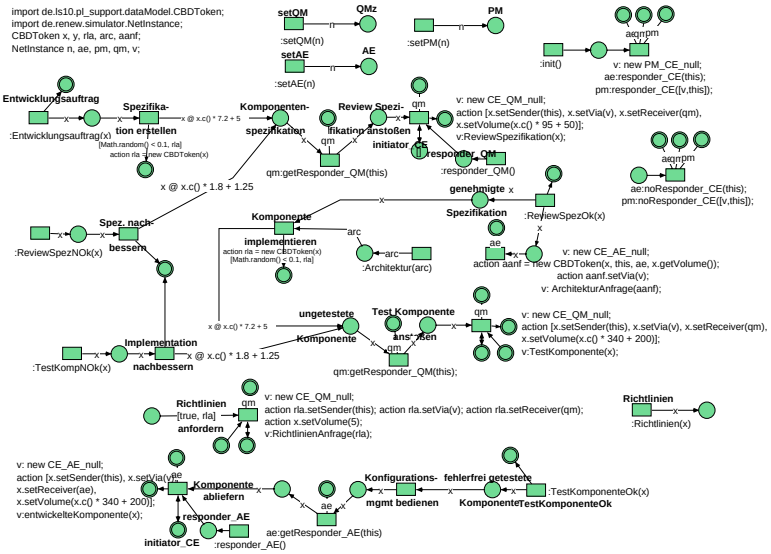


Abbildung A.3: Standort *Komponentenentwicklung* im Simulationsmodell

Aktivität	**Zeitbedarf**	**Volumen**
Spezifikation erstellen	5 - 77 Tage $\hat{=}$ 5 + x.c() $\times$ 7.2	
Review Spezifikation anstoßen		50 - 1000 kB $\hat{=}$ 50 + x.c() $\times$ 95
Komponente implementieren	5 - 77 Tage $\hat{=}$ 5 + x.c() $\times$ 7.2	
Komponente testen lassen		200 - 3600 kB $\hat{=}$ 200 + x.c() $\times$ 340
Komponente abliefern		200 - 3600 kB $\hat{=}$ 200 + x.c() $\times$ 340
Konfigurations-management	1 - 15 Tage $\hat{=}$ 1 + x.c() $\times$ 1.4	
Richtlinien anfragen		5 kB $\hat{=}$ 5

Tabelle A.73: Parametrisierung der Standorte *Komponentenentwicklung-1* bis *-5*

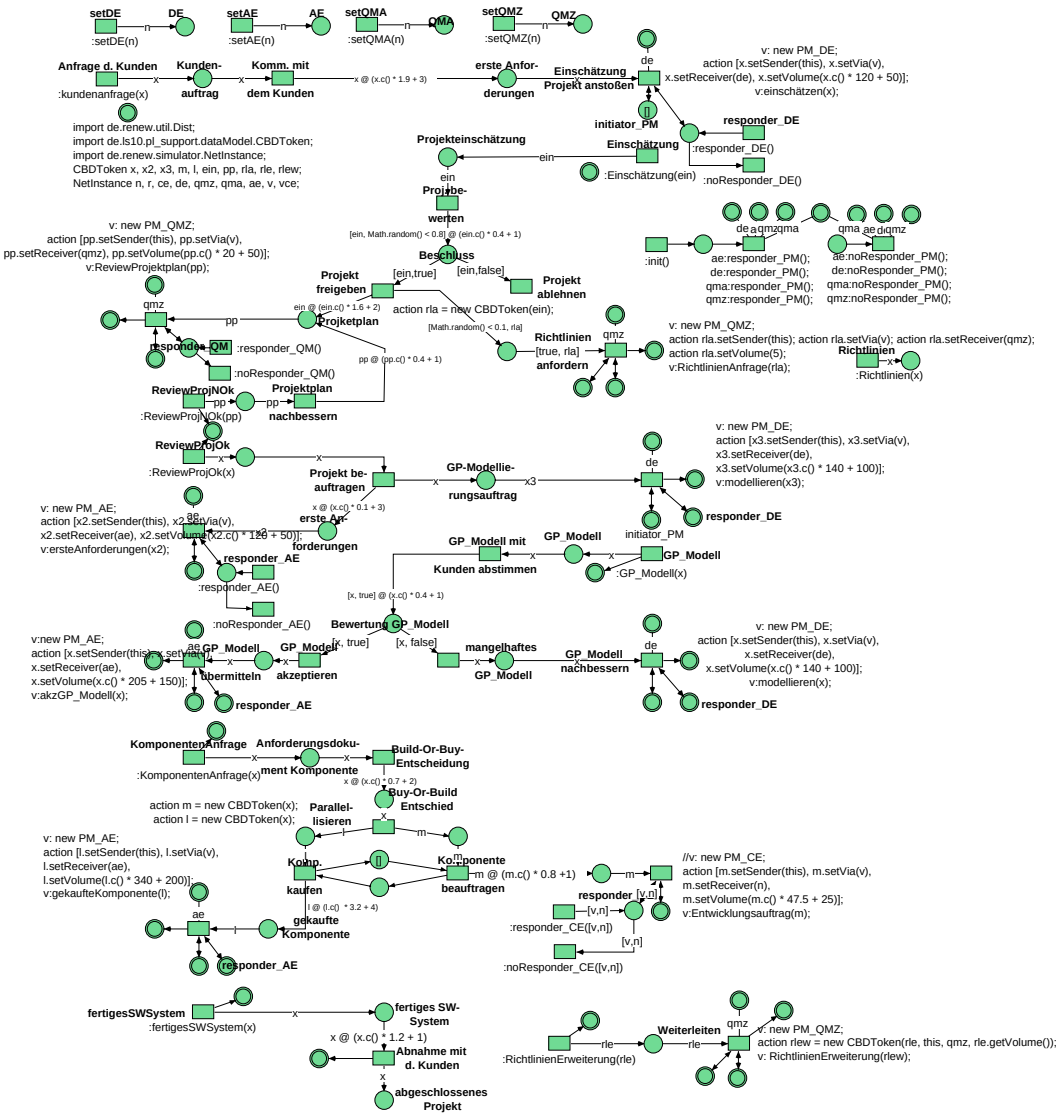


Abbildung A.4: Standort *Projektmanagement* im Simulationsmodell

Aktivität	**Zeitbedarf**	**Volumen**
Kommunikation mit dem Kunden	3 - 22 Tage $\hat{=}$ 3 + x.c() $\times$ 1.9	
Projekteinschätzung anstoßen		50 - 750 kB $\hat{=}$ 50 + x.c() $\times$ 70
Projekt bewerten	1 - 4 Tage $\hat{=}$ 1 + x.c() $\times$ 0.4	
Projekt ablehnen	0 Tage	
Projekt freigeben	2 - 18 Tage $\hat{=}$ 2 + x.c() $\times$ 1.6	
Review Projektplan anstoßen		50 - 250 kB $\hat{=}$ 50 + x.c() $\times$ 20
Projektplan nachbessern	1 - 5 Tage $\hat{=}$ 1 + x.c() $\times$ 0.3	
Geschäftsprozess-modellierung beauftragen		100 - 1500 kB $\hat{=}$ 100 + x.c() $\times$ 140
Projekt beauftragen	3 Tage	
Geschäftsprozessmodell mit Kunden abstimmen	1 - 5 Tage $\hat{=}$ 1 + x.c() $\times$ 0.4	
Geschäftsprozessmodell-Nachbesserung anstoßen		100 - 1500 kB $\hat{=}$ 100 + x.c() $\times$ 140
Geschäftsprozessmodell akzeptieren	0 Tage	
Geschäftsprozessmodell übermitteln		150 - 2200 kB $\hat{=}$ 150 + x.c() $\times$ 205
Build-or-Buy-Entscheidung	2 - 9 Tage $\hat{=}$ 2 + x.c() $\times$ 0.7	
Komponente kaufen	4 - 36 Tage $\hat{=}$ 4 + x.c() $\times$ 3.2	
Komponente beauftragen	1 - 9 Tage $\hat{=}$ 1 + x.c() $\times$ 0.8	25 - 500 kB $\hat{=}$ 25 + x.c() $\times$ 47.5
Abnahme mit dem Kunden	1 - 13 Tage $\hat{=}$ 1 + x.c() $\times$ 1.2	
Richtlinienerweiterungs-vorschlag weiterleiten		0 - 150 kB $\hat{=}$ x.c() $\times$ 15
Richtlinien anfragen		5 kB $\hat{=}$ 5

Tabelle A.74: Parametrisierung des Standortes *Projektmanagement*

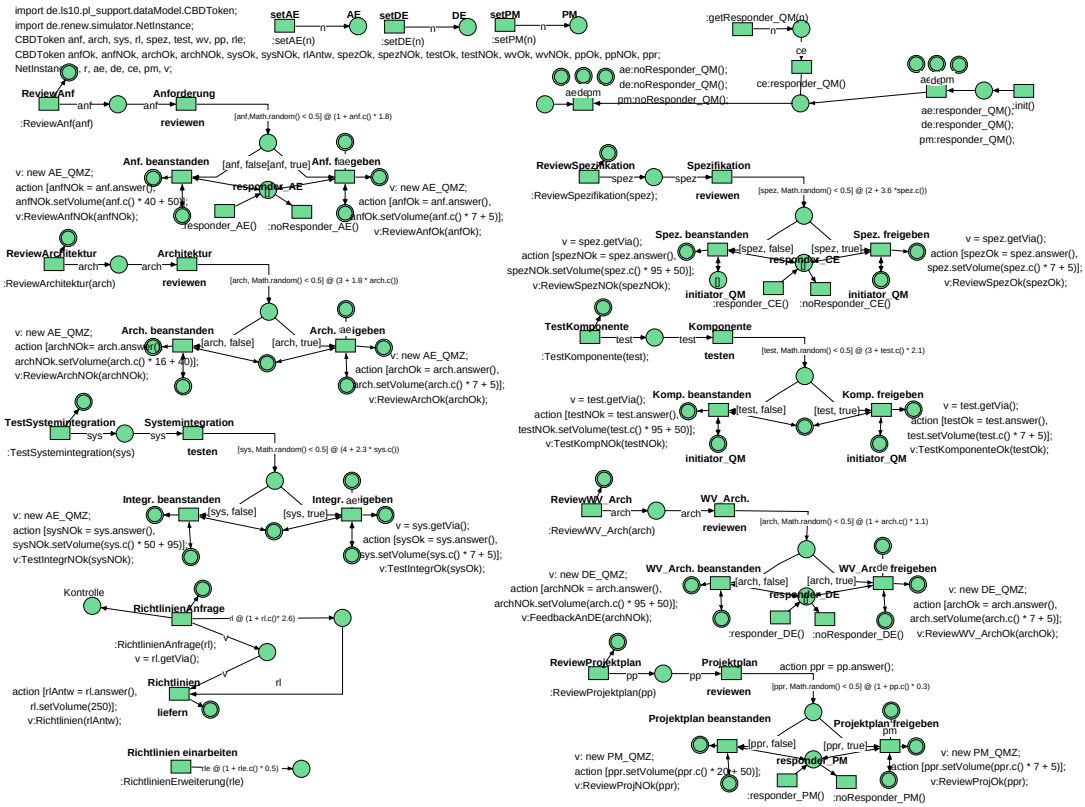


Abbildung A.5: Standort *Qualitätsmanagement-Z* im Simulationsmodell

Aktivität	**Zeitbedarf**	**Volumen**
Projektplan reviewen	1 - 4 Tage $\hat{=}$ 1 + x.c() $\times$ 0.4	
Projektplan beanstanden		50 - 250 kB $\hat{=}$ 50 + x.c() $\times$ 20
Projektplan freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Wiederverwendungs-architektur reviewen	1 - 12 Tage $\hat{=}$ 1 + x.c() $\times$ 1.1	
Wiederverwendungs-architektur beanstanden		50 - 1000 kB $\hat{=}$ 50 + x.c() $\times$ 95
Wiederverwendungs-architektur freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Anforderungsdokument reviewen	1 - 19 Tage $\hat{=}$ 1 + x.c() $\times$ 1.8	
Anforderungsdokument beanstanden		50 - 450 kB $\hat{=}$ 50 + x.c() $\times$ 40
Anforderungsdokument freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Architektur reviewen	3 - 21 Tage $\hat{=}$ 3 + x.c() $\times$ 1.8	
Architektur beanstanden		40 - 205 kB $\hat{=}$ 40 + x.c() $\times$ 16
Architektur freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Systemintegration testen	4 - 27 Tage $\hat{=}$ 4 + x.c() $\times$ 2.3	
Systemintegration beanstanden		50 - 1000 kB $\hat{=}$ 50 + x.c() $\times$ 95
Systemintegration freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Spezifikation reviewen	2 - 38 Tage $\hat{=}$ 2 + x.c() $\times$ 3.6	
Spezifikation beanstanden		50 - 1000 kB $\hat{=}$ 50 + x.c() $\times$ 95
Spezifikation freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Komponente testen	3 - 24 Tage $\hat{=}$ 3 + x.c() $\times$ 2.1	
Komponente beanstanden		50 - 1000 kB $\hat{=}$ 50 + x.c() $\times$ 95
Komponente freigeben		5 - 75 kB $\hat{=}$ 5 + x.c() $\times$ 7
Richtlinienerweiterungs-vorschlag einarbeiten	1 - 27 Tage $\hat{=}$ 1 + x.c() $\times$ 2.6	
Richtlinien liefern	1 - 6 Tage $\hat{=}$ 1 + x.c() $\times$ 0.5	250 kB $\hat{=}$ 250

Tabelle A.75: Parametrisierung des Standortes *Qualitätsmanagement-Z*

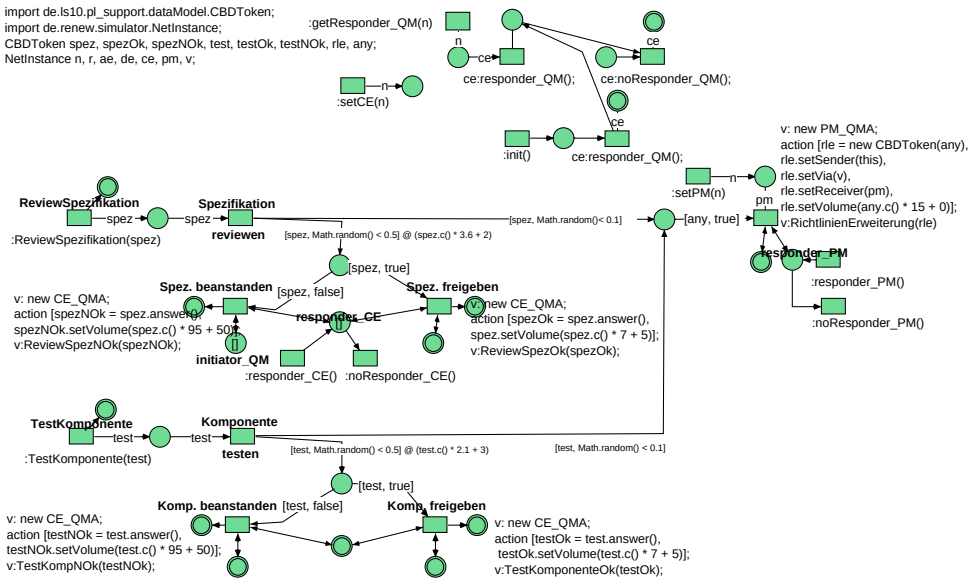


Abbildung A.6: Standort *Qualitätsmanagement-A* der Außenstelle-1 im Simulationsmodell

Der Standort *Qualitätsmanagement-A* der Außenstelle-1 enthält eine Teilmenge der Aktivitäten des zentralen Qualitätsmanagements. Konkret sind dies die Aktivitäten

- Spezifikation reviewen
- Spezifikation beanstanden
- Spezifikation freigeben
- Komponente testen
- Komponente beanstanden
- Komponente freigeben

Zusätzlich enthält der Standort die Aktivität *Richtlinienerweiterungsvorschlag erarbeiten*, deren Volumen in Tabelle A.76 angegeben ist.

Aktivität	**Zeitbedarf**	**Volumen**
Richtlinienerweiterungs-vorschlag erarbeiten		50 - 150 kB $\hat{=}$ x.c() $\times$ 15

Tabelle A.76: Parametrisierung des Standortes *Qualitätsmanagement-A*

Nachfolgend sind weitere Parameter des Simulationsmodells aufgeführt, die Einfluss auf die Kosten der Kommunikationssysteme haben. Dies sind zum einen zwei Kostenmodelle für unterschiedliche Gebührenberechnungen und zum anderen eine Liste aller innerhalb der Beispielprozesslandschaft verwendeten Kommunikationssysteme. Für letztere wird vereinfachend davon ausgegangen, dass jeweils alle Kommunikationskanäle zwischen zwei Standorten zu einem Kommunikationssystem zusammengefasst sind.

Kostenmodelle für Kommunikationssysteme

Kostenmodell a:
Für Standleitungen ist ein monatlicher Festpreis festgelegt, der sich nach der Kapazität und der garantierten Qualität der Leitung richtet. Bei diesem Kostenmodell steht die Leitung den Kommunikationspartnern ohne zusätzliche Gebühren jederzeit zur Verfügung.

Kostenmodell b:
Bei diesem Kostenmodell ist die Nutzungsgebühr abhängig von der Größe des zu übertragenden Volumens, der Kapazität des genutzten Kommunikationskanals, einem festen Preis pro Einheit und der mittleren Dauer einer Einheit. Tabelle A.77 zeigt Beispielwerte, anhand derer die Berechnung der Kosten für dieses Modell verdeutlicht werden.

Variable	**Beschreibung**	**Beispieldaten**
V	zu übertragendes Volumen	(Eingabewert)
K	Kapazität des Kanals	64 kBit/s
P	Preis pro Einheit	0,12 Euro
E	mittlere Dauer einer Einheit	Mittelwert(120s, 15s) = 67,5s

Tabelle A.77: Beispieldaten zum Kostenmodell b

Für dieses Beispiel wird zunächst ein Durchschnittspreis $P_{\oslash}$ berechnet:

$$\begin{aligned} P_{\oslash} &= \text{Preis/Einheit in Euro} \times 60 \text{ s/min / mittlere Dauer einer Einheit in s} \\ &= 0{,}12 \text{ Euro} \times 60 \text{ s/min / } 67{,}5\text{s} \\ &\approx 0{,}11 \text{ Euro/min} \end{aligned}$$

Die Übertragungskosten einer Nachricht einer Größe V lassen sich nun wie folgt abschätzen:

$$K_{\oslash} = P_{\oslash}[\text{Euro}/min] \times \frac{V[kBit]}{K[kBit/s] \times 60[s/min]}$$

Bei beiden Kostenmodellen finden in der Realität zusätzlich anfallende Grundgebühren keine Berücksichtigung.

Kommunikationssystem	**Ausprägung**	**Kosten in Euro**
Anwendungsentwicklung ⇔ Qualitätsmanagement-Z	Ethernet im LAN	0.00
Komponentenentwicklung-1 ⇒ Anwendungsentwicklung	Ethernet im LAN	0.00
Komponentenentwicklung-2 ⇒ Anwendungsentwicklung	ISDN mit Kabelbündung bei RegioCall	2 x 0.11 = 0.22
Komponentenentwicklung-3 ⇒ Anwendungsentwicklung	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Komponentenentwicklung-4 ⇒ Anwendungsentwicklung	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Komponentenentwicklung-5 ⇒ Anwendungsentwicklung	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Komponentenentwicklung-1 ⇔ Qualitätsmanagement-Z	Ethernet im LAN	0.00
Komponentenentwicklung-2 ⇔ Qualitätsmanagement-A	Ethernet im LAN	0.00
Komponentenentwicklung-3 ⇔ Qualitätsmanagement-Z	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Komponentenentwicklung-4 ⇔ Qualitätsmanagement-Z	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Komponentenentwicklung-5 ⇔ Qualitätsmanagement-Z	ISDN mit Kabelbündung bei GlobalCall	2 x 3.63 = 7.26
Domain Engineering ⇔ Anwendungsentwicklung	Ethernet im LAN	0.00
Domain Engineering ⇔ Qualitätsmanagement-Z	Ethernet im LAN	0.00
Projektmanagement ⇔ Anwendungsentwicklung	ISDN bei RegioCall	0.11
Projektmanagement ⇒ Komponentenentwicklung-1	Ethernet im LAN	0.00
Projektmanagement ⇒ Komponentenentwicklung-2	Fax bei RegioCall	0.11
Projektmanagement ⇒ Komponentenentwicklung-3	Fax bei GlobalCall	3.63
Projektmanagement ⇒ Komponentenentwicklung-4	Fax bei GlobalCall	3.63
Projektmanagement ⇒ Komponentenentwicklung-5	Fax bei GlobalCall	3.63

Kommunikationssystem	**Ausprägung**	**Kosten in Euro**
Projektmanagement ⇔ Domain Engineering	Ethernet im LAN	0.00
Projektmanagement ⇔ Qualitätsmanagement-Z	Ethernet im LAN	0.00
Qualitätsmanagement-A ⇒ Projektmanagement	ISDN bei RegioCall	0.11

Tabelle A.78: Kommunikationssysteme der Beispielprozesslandschaft

Tabelle A.79 zeigt abschließend ausgewählte Initiator-/Responder-Erfüllungsraten der modifizierten Prozesslandschaft, die in Abschnitt 4.2.2.2 nicht näher diskutiert werden, da deren Analyse eng mit der der Prozessauslastung zusammenhängt.

Orte	**Min**	**Max**	**erwartungstr. Mittelwert**	**Standard-abweichung**
Anwendungsentwicklung ⇒ Qualitätsmanagement-Z	10,53	43,94	24,25	9,40
Domain Engineering ⇒ Qualitätsmanagement-Z	11,11	53,85	28,43	8,96
Projektmanagement ⇒ Qualitätsmanagement-Z	13,04	58,06	28,33	8,72

Tabelle A.79: Initiator-/Responder-Erfüllungsraten der modifizierten Beispielprozesslandschaft

Literaturverzeichnis

[ABK95] M. Adler, J. W. Byers und R. M. Karp. Parallel Sorting With Limited Bandwidth. In *Proceedings of the 7th Annual ACM Symposium on Parallel Algorithms and Architectures*, Seiten 129–136, Santa Barbara, California, 1995. ACM Press.

[ade99] adesso AG, 1999. `http://www.adesso.de/de/solutions/leusmart/index.html`, Stand 2001.

[AF98] P. Allen und S. Frost. *Component-based Development for Enterprise Systems - Applying the Select Perspective*. Cambridge University Press, 1998.

[AISJ77] C. A. Alexander, S. Ishikawa, M. Silverstein und M. Jacobson. *The Timeless Way of Building*. Oxford University Press, New York, 1977.

[Ale79] C. A. Alexander. *The Timeless Way of Building*. Oxford University Press, New York, 1979.

[Alf03] K. Alfert. *Vitruv: Specifying temporal Aspects of Multimedia Presentations*. Dr. rer. nat. Dissertation, Universität Dortmund, Fachbereich Informatik, Lehrstuhl für Software-Technologie, 2003.

[Amb98] S. W. Ambler. *Process Patterns – Building Large-Scale Systems Using Object Technology*. Cambridge University Press, 1998.

[AP97] G. Abeysinghe und K. Phalp. Combining process modelling methods. *Information and Software Technology*, 39(2):107–124, 1997.

[Bal82] H. Balzert. *Die Entwicklung von Software-Systemen, Prinzipien, Methoden, Sprachen, Werkzeuge*. Bibliographisches Institut, 1982.

[Bal96] H. Balzert. *Lehrbuch der Software-Technik – Software-Entwicklung*. Spektrum Akademischer Verlag, 1996.

[Bal01] H. Balzert. *UML kompakt*. Spektrum Akademischer Verlag, 2001.

[Bau96] B. Baumgarten. *Petri-Netze - Grundlagen und Anwendungen*. Spektrum Akademischer Verlag, 2. Auflage, 1996.

[BBD+99] A. Battke, A. Borusan, J. Dehnert, H. Ehrig, C. Ermel, M. Gajewski, K. Hoffmann, B. Hohberg, G. Juhas, S. Lemke, A. Martens, J. Padberg, W. Reisig, T. Vesper, H. Weber und M. Weber. Initial Realization of the Petri Net Baukasten, 1999. Technischer Bericht, Humbold Universität Berlin, `http://www.informatik.hu-berlin.de/PNT/pnt-public.html`, Stand September 2002.

[BD96] A. Bäumker und W. Dittrich. Fully Dynamic Search Trees for an Extension of the BSP Model. In *Proceedings of the 8th Annual ACM Symposium on Parallel Algorithms and Architectures*, Seiten 233–242, Pagua, Italy, Juni 1996. ACM Press.

[BGKK99] T. Braun, M. Günter, M. Kasumi und I. Khalil. Virtual Private Network Architecture. Interner Bericht, 1999. `http://citeseer.nj.nec.com/braun99virtual.html`, Stand September 2002.

[BJK00] F. Bayer, S. Junginger und H. Kühn. A Business Process-Oriented Methodology for Developing E-Business Applications. In U. F. Baake, R. N. Zobel und M. Al-Akaidi, Hrsg., *Proceedings of the 7th European Concurrent Engineering Conference (ECEC'2000)*, Seiten 32–40, Leicester, UK, April 2000. SCS Press.

[BJSW01] S. Becker, D. Jäger, A. Schleicher und B. Westfechtel. A Delegation Based Model for Distributed Software Process Management. In V. Ambriola, Hrsg., *Software Process Technology – Proceedings of the 8th European Workshop on Software Process Technology EWSPT*, Seiten 130–144, Witten, Germany, 2001. Springer Verlag. Erschienen als Lecture Notes in Computer Science 2077.

[BOC02] BOC GmbH, 2002. `http://www.boc-eu.com/german/aktuelles/index.shtml`, Stand September 2002.

[BPS+99] R. Bastide, P. Palanque, O. Si, D.-H. Le und D. Navarre. Petri-Net Based Behavioural Specification of CORBA Systems. In S. Donatelli und J. Kleijn, Hrsg., *Proceedings of the 20th International Conference on Applications and Theory of Petri Nets APTN'99*, Seiten 66–85, Williamsburg, VA, USA, 1999. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1639.

[Bro02] C. Brockmann. Werkzeugunterstützte Modellierung von Prozesslandschaften auf verschiedenen Abstraktionsebenen. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, Lehrstuhl für Software-Technologie, April 2002.

[BRS95] J. Becker, M. Rosemann und R. Schütte. Grundsätze ordnungsmäßiger Modellierung. *Wirtschaftsinformatik*, 5(37):435–445, Oktober 1995.

[BS89] I. N. Bronstein und K. A. Semendjajew. Taschenbuch der Mathematik, 1989.

[BT98] G. A. Bolcer und R. N. Taylor. Advanced Workflow Management Technologies. *Software Process: Improvement and practice*, 4(3):125–171, 1998.

[CDP95] D. C. Carr, A. Dandekar und D. E. Perry. Experiments in Process Interface Descriptions, Visualizations and Analyses. In W. Schäfer, Hrsg., *Software Process Technology – Proceedings of the 4th European Workshop on Software Process Technology EWSPT*, Seiten 119–137, Noordwijkerhout, The Netherlands, April 1995. Springer Verlag. Erschienen als Lecture Notes in Computer Science 913.

[CFJ98] R. Conradi, A. Fugetta und M. L. Jaccheri. Six Theses on Software Process Research. In V. Gruhn, Hrsg., *Software Process Technology – Proceedings of the 6th European Workshop on Software Process Technology EWSPT*, Seiten 100–104, Weybridge, UK, September 1998. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1487.

[CFS01] F. Cattaneo, A. Fugetta und D. Sciuto. Pursuing coherence in software process assessment and improvement. *Software process improvement and practice*, 6(1):3–22, März 2001.

[Chr92] G. Chroust. *Modelle der Software-Entwicklung*. Oldenbourg Verlag, 1992.

[CKO92] B. Curtis, M. I. Kellner und J. Over. Process Modeling. *Communications of the ACM*, 35(9):75–90, 1992.

[Col02] J. Coldewey. Agile Entwicklung Web-basierter Systeme. *Wirtschaftsinformatik*, 44(3):237–248, 2002.

[CPN00] CPN group. Petri Nets World – Tools and Software, 2000. Universität Aarhus, Dänemark, `http://www.daimi.au.dk/PetriNets/tools/`, Stand September 2002.

[DD98] O. Demirors und E. Demirors. Software Process Improvement in a Small Organization: Difficulties and Suggestions. In V. Gruhn, Hrsg., *Software Process Technology – Proceedings of the 6th European Workshop on Software Process Technology EWSPT*, Seiten 1–12, Weybridge, UK, September 1998. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1487.

[DEA98] S. Dami, J. Estublier und M. Amiour. APEL: a Graphical Yet Executable Formalism for Process Modeling. *Automated Software Engineering*, 5(1):61–96, 1998.

[Dei00] W. Deiters. Information Gathering and Process Modeling in a Petri Net Based Approach. In W. M. van der Aalst, J. Desel und A. Oberweis, Hrsg., *Business Process Management – Models, Techniques, and Empirical Studies*, Seiten 274–288. Springer Verlag, 2000. Erschienen als Lecture Notes in Computer Science 1806.

[DF96] C. Debou und N. Fuchs. AMI: Ein quantitativer Ansatz für Software-Projekt- und Prozessmanagement. In C. Ebert und R. Dumke, Hrsg., *Software-Metriken in der Praxis*, Seiten 181–185. Springer Verlag, 1996.

[DKW99] J.-C. Derniame, A. B. Kaba und B. Warboys. The Software Process: Modelling and Technology. In J.-C. Derniame, A. B. Kaba und D. Wastell, Hrsg., *Software Process: Principles, Methodology, and Technology*, Seiten 1–13. Springer Verlag, 1999. Erschienen als Lecture Notes in Computer Science 1500.

[Emm00] W. Emmerich. Software Engineering and Middleware: A Roadmap. In A. Finkelstein, Hrsg., *The Future of Software Engineering – 22nd International Conference On Software Engineering 2000*, Seiten 117–129, ICSE, Toronto, Kanada, 2000. ACM Press.

[FH93] P. H. Feiler und W. S. Humphrey. Software Process Development and Enactment: Concepts and Definitions. In *Proceedings of the Second International Conference on the Software Process*, Seiten 28–40, Berlin, Germany, Februar 1993. IEEE Computer Society Press.

[FKN+92] A. Finkelstein, J. Kramer, J. Nuseibeh, L. Finkelstein und M. Goedicke. Viewpoints: A Framework for Integrating Multiple Perspectives in System Development. *International Journal of Software Engineering and Knowledge Engineering*, 2(1):31–57, 1992.

[FKT00] I. Fischer, M. Koch und G. Taentzer. Local Views on Distributed Systems and their Communications. In H. Ehrig, G. Engels, H.-J. Kreowski und G. Rozenberg, Hrsg., *Proceedings of the 6th International Workshop on Theory and Application of Graph Transformation (TAGT'98)*, Seiten 164–178, Paderborn, Germany, November 2000. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1764.

[FR99] X. Franch und R. M. Ribo. Using UML for Modelling the Static Part of a Software Process. In R. France und B. Rumpe, Hrsg., *Proceedings of the 2th Unified Language Conference UML'99*, Seiten 292–307, Colorado, USA, Oktober 1999. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1723.

[FRS+01] M. Friedewald, H. D. Rombach, P. Stahl, M. Broy, S. Hartkopf, S. Kimpeler, K. Kohler, R. Wucher und P. Zoche. Softwareentwicklung in Deutschland. *Informatik Spektrum*, 24(2):81–90, April 2001.

[FS96] A. Finkelstein und I. Sommerville. The Viewpoints FAQ. *Software Engineering Journal*, 11(1):2–4, 1996.

[Fug00] A. Fugetta. Software Process: A Roadmap. In A. Finkelstein, Hrsg., *The Future of Software Engineering, 22nd International Conference on Software Engineering*, Seiten 25–33, ICSE, Toronto, Kanada, 2000. ACM Press.

[GE01] M. Gajewski und H. Ehrig. The «Petri Net Baukasten»: An Overview. In H. Ehrig, G. Juhas, J. Padberg und G. Rozenberg, Hrsg., *Unifying Petri Nets – Advances in Petri Nets*, Seiten 26–53. Springer Verlag, 2001. Erschienen als Lecture Notes in Computer Science 2128.

[GEMT00] M. Goedicke, B. Enders, T. Meyer und G. Taentzer. ViewPoint-Oriented Software Development: Tool Support for Integrating Multiple Perspectives by Distributed Graph Transformation. In S. Graf und M. Schwartzbach, Hrsg., *Proceedings of the 6th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Seiten 43–47, Berlin, Germany, März/April 2000. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1785.

[Gen86] H. J. Genrich. Predicate/Transition Nets. In G. Goos und J. Hartmanis, Hrsg., *Petri Nets: Central Models and Their Properties – Advances in Petri Nets 1986, Part I, Proceedings of an Advanced Course*, Seiten 207–247, Bad Honnef, Germany, September 1986. Springer Verlag. Erschienen als Lecture Notes in Computer Science 254.

[GF01] Arbeitskreis „Begriffe und Konzepte der Vorgehensmodellierung“ GI-Fachgruppe 5.11. Begriffssammlung Vorgehensmodelle, 2001. `http://www.vorgehensmodelle.de/Giak/arbeitskreise/vorgehensmodelle/themenbereiche/prinzipMethodeWerkzeug.html`, Stand: September 2002.

[GHJV94] E. Gamma, R. Helm, R. Johnson und J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, 1994.

[GMT99] M. Goedicke, T. Meyer und G. Taentzer. ViewPoint-oriented Software Development by Distributed Graph Transformation: Towards a Basis for Living with Inconsistencies. In *Proceedings of the 4th IEEE International Symposium on Requirements Engineering*, Seiten 92–99, Limerick, Ireland, Juni 1999. IEEE Computer Society Press.

[Gre88] I. Greif. *Computer-Supported Cooperative Work – A Book of Readings*. Morgan Kaufmann, 1988.

[Gru91] V. Gruhn. *Validation and Verification of Software Process Models*. Dr. rer. nat. Dissertation, Universität Dortmund, Juni 1991.

[GT00] V. Gruhn und A. Thiel. *Komponentenmodelle*. Addison Wesley, 2000.

[GW99a] V. Gruhn und U. Wellen. Software Process Landscaping – An Approach to Structure Complex Software Processes. In *International Process Technology Workshop (IPTW) Proceedings*, 1999.

[GW99b] V. Gruhn und U. Wellen. Software Support for Distributed Business Processes. In *Asia-Pacific Software Engineering Conference*, Seiten 200–205, Takamatsu, Japan, 1999. IEEE Computer Society Press.

[GW00a] V. Gruhn und U. Wellen. Process Landscaping – eine Methode zur Geschäftsprozessmodellierung. *Wirtschaftsinformatik*, 4:297–309, August 2000.

[GW00b] V. Gruhn und U. Wellen. Structuring Complex Software Processes by „Process Landscaping“. In R. Conradi, Hrsg., *Software Process Technology – Proceedings of the 7th European Workshop on Software Process Technology EWSPT*, Seiten 138–149, Kaprun, Austria, Februar 2000. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1780.

[GW01a] V. Gruhn und U. Wellen. Analysing a process landscape by simulation. *The Journal of Systems and Software*, 59:333–342, 2001.

[GW01b] V. Gruhn und U. Wellen. Process Landscaping – Modelling Distributed Processes and Proving Properties of Distributed Process Models. In H. Ehrig, G. Juhas, J. Padberg und G. Rozenberg, Hrsg., *Unifying Petri Nets – Advances in Petri Nets*, Seiten 103–125. Springer Verlag, 2001. Erschienen als Lecture Notes in Computer Science 2128.

[GW02] V. Gruhn und U. Wellen. Autonomies in a Software Process Landscape. Interner Bericht 120, Universität Dortmund, Lehrstuhl Software-Technologie, Januar 2002.

[Hay00] N. Hayes. Work-arounds and Boundary Crossing in a High Tech Optronics Company: The Role of Co-operative Workflow Technologies. *Computer Supported Cooperative Work (CSCW) – The Journal of Collaborative Computing*, 9(3):435–455, 2000.

[HB96] T. Hess und L. Brecht. *State of the Art des Business Process Redesign: Darstellung und Vergleich bestehender Methoden*. Gabler Verlag, 2. Auflage, 1996.

[HC93] M. Hammer und J. Champy. *Reengineering the Corporation – A Manifesto for Business Revolution*. Harper Collins Publishers, 1993.

[Höd98] M. Höderath. IV-Strategie als Voraussetzung für IV-Controlling am Beispiel eines Energieversorgers. *Controlling*, 10(2):72–79, März 1998.

[Hey95] M. Heym. *Prozess- und Methoden-Management für Informationssysteme*. Springer Verlag, 1995.

[HH99] T. Hess und V. Herwig. Portale im Internet. *Wirtschaftsinformatik*, 41(6):551–553, 1999.

[HHL98] T. Herrmann, M. Hoffmann und K.-U. Loser. Sozio-orientierte und semi-strukturierte Modellierung mit SeeMe. In *Proceedings der Fachtagung MobIS'98, Rundbrief des GI-Fachausschusses 5.2, Wirtschaftsinformatik*, Seiten 15–22, 1998.

[HK00] P. Herrmann und H. Krumm. A Framework for Modeling Transfer Protocols. *Computer Networks*, 34(2):317–337, 2000.

[HKL02] T. Herrmann, G. Kunau und K.-U. Loser. Sociotechnical Walkthrough – ein methodischer Beitrag zur Gestaltung soziotechnischer Systeme. 2002.

[HKM01] W. Hasselbrink, A. Koschel und A. Mester. Basistechnologien für die Entwicklung von Internet-Portalen. In A. Heuer, F. Leymann und D. Priebe, Hrsg., *Datenbanksysteme in Büro, Technik und Wissenschaft, 9. GI-Fachtagung Oldenburg*, Seiten 517–526. Springer Verlag, 2001.

[HMF92] W. Hesse, G. Merbeth und R. Frölich. *Software-Entwicklung – Vorgehensmodelle, Projektführung,Produktverwaltung, Handbuch der Informatik*. Band 5.3. Oldenbourg Verlag, 1992.

[Hoa85] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[IDS01] IDS Scheer AG, 2001. `http://www.ids-scheer.com/de/`, Rubrik Produkte, Stand September 2002.

[Int01] IntraWare AG, 2001. `http://www.intraware.de/`, Rubrik GPM, Stand September 2002.

[IPB01] P. Isacsson, G. Pedersen und S. Bang. Accelerating CMM-based improvement programs: the accelerator model and method with experiences. *Software process improvement and practice*, 6(1):23–34, März 2001.

[JBR98] I. Jacobson, G. Booch und J. Rumbaugh. *The Unified Software Development Process*. Addison Wesley, 1998.

[Jen97] K. Jensen. *Coloured Petri Nets – Basic Concepts, Analysis Methods and Practical Use*, Band 1. Springer Verlag, 2. Auflage, 1997.

[KBO96] M. I. Kellner, L. Briand und J. W. Over. A Method for Designing, Defining and Evolving Software Processes. In *Proceedings of the 4th International Conference on the Software Process*, Seiten 37–48, Brighton, UK, Dezember 1996. IEEE Computer Society Press.

[Kee97] P. G. W. Keen, Hrsg. *The Process Edge*. Harvard Business School Press, 1997.

[KJKP99] H. Kühn, S. Junginger, D. Karagiannis und C. Petersen. Metamodellierung im Geschäftsprozessmanagement: Konzepte, Erfahrungen und Potentiale. In J. Desel, K. Pohl und A. Schürr, Hrsg., *Modellierung '99 – Workshop der Gesellschaft für Informatik, März 1999 in Karlsruhe*, Seiten 75–90. Teubner Verlag, 1999.

[Kor00] D. S. Koreimann. *Grundlagen der Software-Entwicklung*. Oldenbourg Verlag, 3. Auflage, 2000.

[Kra96] H. Krallmann. *Systemanalyse im Unternehmen*. Oldenbourg Verlag, 2. Auflage, 1996.

[Kru84] H. Krumm. *Spezifikation, Implementierung und Verifikation von Kommunikationsdiensten für verteilte DV-Systeme*. Dr. rer. nat. Dissertation, Universität Karlsruhe, Juni 1984.

[KSK+94] P. Kuvaja, J. Similä, L. Krzanik, A. Bicego, S. Saukkonen und G. Koch. *Software Process Assessment and Improvement: the BOOTSTRAP Approach*. Blackwell Publishers, Oxford, 1994.

[Kum00] O. Kummer. Renew – The Reference Net Workshop, 2000. Tool Demonstrations, 21st International Conference on Application and Theory of Petri Nets, `http://www.renew.de`, Stand Juli 2002.

[KW99] O. Kummer und F. Wienberg. Renew – The Reference Net Workshop. *Petri Net Newsletter*, 56:12–16, 1999.

[KW02] C. Kopka und U. Wellen. Role-based Views to Approach Suitable Software Process Models for the Development of Multimedia Systems. In *Proceedings of the IEEE Forth International Symposium on Multimedia Software Engineering (MSE'2002)*, Seiten 140–147, Newport Beach, California, Dezember 2002. IEEE Computer Society.

[KWD00] O. Kummer, F. Wienberg und M. Duvigneau. Renew – User Guide, Release 1.4, November 2000, 2000. Universität Hamburg, `http://www.renew.de`, Stand Juli 2001.

[LK91] A. M. Law und W. D. Kelton. *Simulation, Modeling & Analysis*. McGraw-Hill, 2. Auflage, 1991.

[Lon93] J. Lonchamp. A Structured Conceptual and Terminological Framework for Software Process Engineering. In *Proceedings of the Second International Conference on the Software Process*, Seiten 41–53, Berlin, Germany, Februar 1993. IEEE Computer Society Press.

[LSS99] S. Leinenbach, C. Seel und A.-W. Scheer. Metamodellierung im Geschäftsprozessmanagement: Konzepte, Erfahrungen und Potentiale. In J. Desel, K. Pohl und A. Schürr, Hrsg., *Modellierung '99 – Workshop der Gesellschaft für Informatik, März 1999 in Karlsruhe*, Seiten 11–26. Teubner Verlag, 1999.

[LSW97] P. Langner, C. Schneider und J. Wehler. Prozessmodellierung mit ereignisgesteuerten Prozeßketten (EPKs) und Petrinetzen. *Wirtschaftsinformatik*, 39(5):479–489, 1997.

[MENW99] T. Menzies, S. Easterbrook, B. Nuseibeh und S. Waugh. An empirical investigation of multiple viewpoint reasoning in requirements engineering. In *Proceedings of the 4th International Symposium on Requirements Engineering (RE'99)*, Seiten 100–110, Limerick, Ireland, Juni 1999. IEEE Computer Society Press.

[MID01] MID, 2001. `http://www.mid.de/de/innovator/`, Stand September 2002.

[OR86] A. M. Ould und C. Roberts. Modeling iteration in the software process. In M. Dowson, Hrsg., *Proceedings of the 3th International Software Process Workshop*, Seiten 101–104, Beckenridge, Colorado, USA, November 1986. IEEE Computer Society Press.

[Ort83] B. Orth. Grundlagen des Messens. In H. Fegert und J. Bredenkamp, Hrsg., *Messen und Testen*, Band 4, Seiten 136–180. Verlag für Psychologie, 1983.

[Pad98] J. Padberg. Abstract Petri Nets as a Uniform Approach to High-Level Petri Nets. In J. L. Fiadeiro, Hrsg., *Proceedings of the 13th European Workshop on Algebraic Development Techniques, WADT'98*, Seiten 240–259, Lissabon, Portugal, April 1998. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1589.

[Poh00] A. Pohlmann. Visualisierung und Simulation von Prozeßlandschaften – Die lokale Sichtweise. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, Lehrstuhl für Software-Technologie, Mai 2000.

[Pro01] Projektgruppe Palermo. Endbericht der Projektgruppe Palermo. Interner Bericht 109, Universität Dortmund, Fachbereich Informatik, Lehrstuhl für Software-Technologie, Februar 2001.

[PW94] W. Pfeiffer und E. Weiß. *Lean-Management – Grundlagen der Führung und Organisation lernender Unternehmen*. Erich Schmidt Verlag, 1994.

[PWCC94] M. Paulk, C. Weber, B. Curtis und M. Chrissis, Hrsg. *The Capability Maturity Model*. Addison Wesley, 1994.

[RHV00] D. Raffo, W. Harrison und J. Vandeville. Coordinating models and metrics to manage software projects. *Software process improvement and practice*, 5(2-3):159–168, Juni/September 2000.

[RJ00] L. Rising und N. S. Janoff. The Scrum Software Development Process for Small Teams. *IEEE Software*, 17(4):26–32, Juli/August 2000.

[Ros96] M. Rosemann. *Komplexitätsmanagement in Prozessmodellen*. Gabler Verlag, 1996.

[RVM99] D. M. Raffo, J. V. Vandeville und R. Martin. Software Process Simulation to Achieve Higher CMM Levels. *Journal of Systems and Software*, 46(2-3):163–172, 1999.

[SG87] M. L. G. Shaw und B. R. Gaines. An Interactive Knowledge-Elicitation Technique using Personal Construct Technology. In A. L. Kidd, Hrsg., *Kowledge Acquisition for Expert Systems – A Practical Handbook*, Seiten 109–136. Plenum Press, 1987.

[SHO95] S. M. Sutton, D. Heimbigner und L. J. Osterweil. APP/L: A Language for Software Process Programming. *ACM Transactions on Software Engineering and Methodology*, 4(3):221–286, 1995.

[Sim96] J.-M. Simon. SPICE: Overview for software process improvement. *Journal of Systems Architecture*, 42(8):633–641, 1996.

[Smi98] E. Smith. Principles of High-Level Net Theory. In W. Reisig und G. Rozenberg, Hrsg., *Lectures on Petri Nets I: Basic Models – Advances in Petri Nets*, Seiten 174–210. Springer Verlag, 1998. Erschienen als Lecture Notes in Computer Science 1491.

[SO97] S. M. Sutton und L. J. Osterweil. The Design of a Next-Generation Process Language. In M. Jazayeri und H. Schauer, Hrsg., *Proceedings of the Joint 6th European Software Engineering Conference and the 5th ACM SIGSOFT Symposium on the Foundation of Software Engineering*, Seiten 142–158, Zürich, Switzerland, September 1997. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1301.

[SOSW98] B. Staudt Lerner, L. J. Osterweil, S. M. Sutton Jr. und A. Wise. Programming Process Coordination in Little-JIL. In V. Gruhn, Hrsg., *Software Process Technology – Proceedings of the 6th European Workshop on Software Process Technology EWSPT*, Seiten 127–131, Weybridge, UK, September 1998. Springer Verlag. Erschienen als Lecture Notes in Computer Science 1487.

[Spi92] J. M. Spivey. *The Z Notation: A Reference Manual*. Prentice Hall, 2. Auflage, 1992.

[Stö01a] H. Störrle. Describing Process Patterns with UML. In V. Ambriola, Hrsg., *Software Process Technology – Proceedings of the 8th European Workshop on Software Process Technology EWSPT*, Seiten 173–181, Witten, Germany, 2001. Springer Verlag. Erschienen als Lecture Notes in Computer Science 2077.

[Stö01b] M. Störzel. Simulation verteilter Prozesslandschaften. Diplomarbeit, Universität Dortmund, Fachbereich Informatik, Lehrstuhl für Software-Technologie, Oktober 2001.

[SW02a] M. Störzel und U. Wellen. Modelling and Simulation of Communication between Distributed Processes. In M. Al-Akaidi, Hrsg., *Proceedings of the Fourth Middle East Symposium on Simulation and Modelling (MESM'2002)*, Seiten 156–160, Sharjah, U.A.E., September 2002. SCS European Publishing House. ISBN: 90-77039-09-0.

[SW02b] M. Störzel und U. Wellen. Tool Support for Distributed Management of Simulation Models and Evaluation Data. In A. Verbraeck und W. Krug, Hrsg., *Proceedings of the 14th European Simulation Symposium*, Seiten 387–392, Dresden, Germany, Oktober 2002. SCS European Publishing House. ISBN: 3-936150-21-4.

[SWR+00] P. Stahl, R. Wucher, H. D. Rombach, S. Hartkopf, K. Kohler, M. Friedwald, S. Kimpeler, P. Zoche, M. Broy und I. Krüger. Analyse und Evaluation der Softwareentwicklung in Deutschland. Studie, Bundesministerium für Bildung und Forschung BMBF, Nürnberg, Dezember 2000. `http://www.iid.de`, Stand September 2002.

[TE00] G. Taentzer und H. Ehrig. Semantics of Distributed System Specifications based on Graph Transformation, 2000. GI-Jahrestagung 2000, Workshop „Rigorose Entwicklung software-intensiver Systeme".

[VB96] G. Vossen und J. Becker, Hrsg. *Geschäftsprozeßmodellierung und Workflow-Management*. Thompson Publishing, 1996.

[WC01] A. I. Wang und L. Chunnian. Process Support for Mobile Work across Heterogeneous Systems. In V. Ambriola, Hrsg., *Software Process Technology – Proceedings of the 8th European Workshop on Software Process Technology EWSPT*, Seiten 117–129, Witten, Germany, 2001. Springer Verlag. Erschienen als Lecture Notes in Computer Science 2077.

[WSvW+00] T. Weitzel, S. Son, F. v. Westarp, P. Buxmann und W. König. Wirtschaftlichkeitsanalyse von Kommunikationsstandards – eine Fallstudie am Beispiel von X.500 Directory Services mit der Siemens AG. SFB 403 AB-00-05, Institut für Wirtschaftsinformatik, J. W. Goethe-Universität, Frankfurt am Main,

2000. `http://caladan.wiwi.uni-frankfurt.de/IWI/projectb3/deu/publikat/x500/index.htm`, Stand September 2002.

[Zie96] J. Ziegler. Rechnerunterstützung für kooperative Arbeit – Computer Supported Cooperative Work. In H.-J. Bullinger und H. J. Warnecke, Hrsg., *Ein Handbuch für das moderne Management*, Seiten 680–690. Springer Verlag, 1996.

[ZL01] K. Zamli und P. Lee. Taxonomie of Process Modeling Languages. In *Proceedings of ACS/IEEE International Conference on Computer Systems and Applications*, Seiten 435–437, Beirut, Libanon, Juni 2001. IEEE Computer Society Press.

[ZPK00] B. P. Zeigler, H. Praehofer und T. G. Kim. *Theory of Modeling and Simulation – Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, 2. Auflage, 2000.

Abbildungsverzeichnis

Tabellenverzeichnis