JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

# Analyse und Vergleich von ausgewählten Ansätzen zur Wiederverwendung von Prozessmodellen

**DIPLOMARBEIT**

zur Erlangung des akademischen Grades eines
Magisters der Sozial- und Wirtschaftswissenschaften
(Mag.rer.soc.oec.)


am Institut für Wirtschaftsinformatik
der Johannes-Kepler-Universität Linz
Abteilung für Data & Knowledge Engineering

Eingereicht von
**SILVIA GROBNER**

Begutachter
o. Univ.-Prof. Dr. Michael Schrefl

Mitbetreuer
Mag. Andreas Bögl


Linz, April 2008

## Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Linz, April 2008

## Danksagung

An dieser Stelle möchte ich mich bei meinen Betreuern Herrn o. Univ.-Prof. Dr. Michael Schrefl und Herrn Mag. Andreas Bögl bedanken, die durch ihre hervorragende fachliche Unterstützung und ihre herausragenden Anregungen wesentlich zur Entstehung der vorliegenden Arbeit beigetragen haben.

Besonderen Dank möchte ich meinem Freund Philipp aussprechen, der wegen seiner motivierenden und unterstützenden Worte während des gesamten Erstellungsprozesses eine gesonderte Erwähnung verdient. Desweiteren bedanke ich mich bei meinen Eltern und meinen Freunden, die mir ebenfalls ermutigend und motivierend zur Seite gestanden haben.

## Abstract

Business processes usually are confronted with steady adaptations triggered by an increasing flexibility, changes in customer needs or by a growing pressure of competition. Such adaptations on processes should be made time and cost effectively.

Holistic researches address the reuse of existing process knowledge in order to learn from best or common practices. Applying different reuse strategies imply several problems such as how to make implicit process knowledge explicit, how to capture that knowledge in a common knowledge base and how to retrieve suitable process solutions and to configure them according to new process requirements.

This work compares three selected reuse approaches that follow a pattern-based, a building block-based and a case-based reasoning approach. Evaluation criteria derived from literature are applied to a common process example. The evaluation results are used to determine strengths and weaknesses of the selected approaches and their prototypical implementations.

## Zusammenfassung

Die in Unternehmen definierten Prozesse unterliegen ständigen Anpassungen, die insbesondere aus den Forderungen nach Flexibilität, Änderungen von Kundenbedürfnissen, aber auch durch den ständig wachsenden Konkurrenzdruck bestimmt werden. Damit unterliegen die Prozessmodelle, die die gesamte Unternehmensstruktur in organisatorischer und ablauforientierter Hinsicht darstellen, dynamischen Einflüssen.

Änderungen an bestehenden Prozessen sollen zeit- und kosteneffektiv durchgeführt werden. Umfangreiche Forschungen beschäftigen sich mit dem Thema der Wiederverwendung von Prozessmodellen, um einerseits von Best oder Common Practices zu lernen und andererseits auf bestehendes Prozesswissen in Form von Prozessmodellen im Zuge des Prozessdesigns zurückzugreifen.

Die Anwendung von Strategien zur Wiederverwendung von bereits vorhandenem Prozesswissen ist mit zahlreichen Problemstellungen verbunden. Dazu zählen wie implizites Prozesswissen in einer Wissensbasis explizit verfügbar gemacht werden kann, wie für bestimmte Problemstellungen adäquate Prozesslösungen gefunden werden und inwiefern geänderte Anforderungen in eine neue Prozesslösung systematisch integriert werden können.

In der vorliegenden Arbeit werden ausgewählte Ansätze der Wiederverwendung vorgestellt. Im Konkreten handelt es sich um Konzepte der musterbasierten, bausteinbasierten und fallbasierten Wiederverwendung von Prozesswissen. Diese Konzepte werden eingehend untersucht, wobei auch kurz auf die prototypische Umsetzung dieser Konzepte eingegangen wird. Anhand aus der Literatur abgeleiteter Kriterien werden die Konzepte und Prototypen bewertet, um Erkenntnisse über Stärken und Schwächen einzelner Ansätze zu identifizieren.

# Inhaltsverzeichnis

## Abbildungsverzeichnis

## Tabellenverzeichnis

# 1 Einleitung

## *1.1 Motivation*

Der Erfolg von Unternehmen ist nicht mehr nur von den angebotenen Produkten oder Dienstleistungen abhängig, sondern auch von der Effizienz und Effektivität der Abläufe im Unternehmen wie diverse Studien beweisen (z.B. [Rohl07], [Seid02], [SeWu02]). Ein Teil des angestrebten Unternehmenserfolgs kann daher dadurch erreicht werden, indem Geschäftsprozesse, das sind die betrieblichen Abläufe in einem Unternehmen, effizient und effektiv durchgeführt werden. Aus diesem Grund wurden unterschiedliche Methoden der Geschäftsprozessmodellierung entwickelt, um die Geschäftsprozesse strukturiert und grafisch darzustellen, z.B. EPK (vgl. [Sche01]), Petri-Netze (vgl. [Baum90]), und die bestehenden Geschäftsprozesse durch methodische Vorgehensweisen zu optimieren (vgl. z.B. [Rose02, 212ff.]. Doch ist die Neuerstellung der Geschäftsprozesse für jedes einzelne Projekt, und somit *„auf der grünen Wiese“* zu modellieren, sehr zeitintensiv und aufwändig [Lang97, 2].

Daher beschäftigen sich immer mehr Autoren mit der Wiederverwendung vorgefertigter Prozesslösungen. Dabei wurden umfangreiche Forschungen über die proaktive und reaktive Wiederverwendung betrieben (vgl. z.B. [Broc03], [BDKK02], [FeLo04], [Schu01]). Ansätze der *proaktiven Wiederverwendung* erstellen *„wiederverwendbare Artefakte vor Eintreten eines konkreten Bedarfs“*, wohingegen Ansätze der *reaktiven Wiederverwendung* existierende Artefakte wiederverwenden, nachdem ein konkreter Bedarf eingetreten ist [BöKS06, 2].

Ein Vertreter der proaktiven Wiederverwendung ist die Referenzmodellierung (vgl. beispielsweise [BeKn02]), dessen Prozesslösungen als Vorlage für die Geschäftsprozessmodellierung dienen [HeHR04] und manuell an das jeweilige Unternehmen angepasst werden können.

Zur reaktiven Wiederverwendung existieren unterschiedliche Klassen, wobei die Ansätze der reaktiven Wiederverwendung den musterbasierten (patternbasierten), bausteinbasierten und fallbasierten Klassen zugeordnet werden können. Musterbasierte Ansätze stellen das Prozesswissen in Form von Prozessmustern dar. Dazu zählen beispielsweise die Ansätze nach Hagen [Hage05], nach Gnatz et al. [GMPR01], nach Coplien [Copl94, Copl95, Copl96], nach Ambler [Ambl98], nach Störrle [Stör01] und nach Noble [Nobl98]). Bausteinbasierte Ansätze machen das Prozesswissen in Form von Prozessbausteinen wiederverwendbar. Dazu zählen unter anderem Rupprecht [Rupp02], Lang [Lang97] und Remme [Remm97]. Fallbasierte Ansätze stellen das Prozesswissen in Form von Fallbeispielen zur Verfügung, die durch fallbasiertes Schließen wiederverwendbar gemacht werden. Dazu zählen unter anderem die Ansätze nach Krampe [Kram98] und Schulze [Schu01].

Ziel der reaktiven Wiederverwendung ist somit Konzepte zur Verfügung zu stellen, die das Prozesswissen in Form von Prozessmodellen nach einem konkreten Bedarf wiederverwendbar machen. Dazu müssen sie die Prozessmodelle für die Wissensbasis, in der sie gespeichert werden, entsprechend aufbereiten und durch Such- und Anpassungsstrategien an eine neue Problemstellung anpassen.

Im Rahmen dieser Untersuchung wird auf Ansätze der reaktiven Wiederverwendung eingegangen und jeweils ein Vertreter der Klassen der Ansätze zur Untersuchung ausgewählt. Es werden die Ansätze nach Hagen, Rupprecht und Krampe untersucht, die in Dissertationen prototypisch umgesetzte Konzepte zur Wiederverwendung entwickelt haben. Es gilt zu

untersuchen inwieweit die Konzepte die Anforderungen nach der Suche und Anpassung unterstützen, und somit Prozesslösungen durch die Konzepte generiert werden können.

## *1.2 Zielsetzung und Lösungsansatz*

Ziel dieser Arbeit ist es, die ausgewählten Modellierungsansätze zur reaktiven Wiederverwendung zu analysieren und zu vergleichen, und sowohl Stärken als auch Schwächen der Konzepte und realisierender Prototypen zu identifizieren. Es werden jeweils Vertreter der unterschiedlichen Klassen der Ansätze der reaktiven Wiederverwendung ausgewählt, und zwar die Ansätze nach Hagen (musterbasierter Ansatz), Rupprecht (bausteinbasierter Ansatz) und Krampe (fallbasierter Ansatz).

Es muss ein Lösungsansatz gefunden werden, wie die Analyse und der Vergleich methodisch durchgeführt werden können, um so die Stärken und Schwächen der untersuchten Ansätze identifizieren zu können. Dies führt zu den Problemen, welche Kriterien die Grundlage für den Vergleich bilden, und welche Ausprägungen für die Bewertung verwendet werden können, um zu einem sinnvollen Ergebnis zu kommen.

Dabei wird nach folgendem Vorgehensmodell vorgegangen, das in Anlehnung an [BoDö01] entwickelt wurde (Abbildung 1). Nach einer Vertiefung in die ausgewählten Modellierungsansätze [Hage05], [Rupp02], [Kram98], werden gemeinsam verwendete Konzepte der reaktiven Wiederverwendung identifiziert und beschrieben. Anschließend wird ein Rahmenbeispiel erstellt, das das Ergebnis nach der Anwendung der einzelnen Ansätze sein soll. Um zu dem Ergebnis gelangen zu können, werden mehrere Prozessmodelle erstellt, die als Grundlage für die reaktive Wiederverwendung dienen. Nach diesen Schritten wird iterativ jeder Ansatz genau im Hinblick auf die reaktive Wiederverwendung untersucht und anhand von Beispielen erklärt. Nachdem die Untersuchung Ansätze abgeschlossen ist, wird ein Kriterienkatalog aus der Literatur abgeleitet und der Vergleich und die anschließende Bewertung auf Basis dieser Kriterien durchgeführt. Die Kriterien werden dem Konzept und dem Prototyp zugeordnet, da ein Prototyp austauschbar ist und das Konzept auch durch einen anderen Prototyp realisiert werden kann.

Die Untersuchung der jeweiligen Ansätze erfolgt nach dem in Abbildung 2 dargestellten Untersuchungsdesign.

Zunächst wird ein Modellierungsproblem (Aufgabenstellung/Zielsetzung der Modellierung) natürlichsprachig formuliert (Abschnitt 1.3.1). Mit anderen Worten, es wird der Prozess, der durch die Anwendung eines Ansatzes der reaktiven Wiederverwendung erreicht werden soll, definiert. Als nächstes wird der zu erwartende Output modelliert (Abschnitt 1.3.2). Dies soll die Prozesslösung sein, die durch die Anwendung der ausgewählten Ansätze erreicht werden soll. Um den Soll-Output mit Hilfe der reaktiven Wiederverwendung zu erreichen, dienen mehrere Prozessmodelle aus Abschnitt 1.4 als Grundlage, auf die während der reaktiven Wiederverwendung zurückgegriffen wird.

Zu Beginn müssen diese Prozessmodelle für die spätere Wiederverwendung in der Wissensbasis dem jeweiligen Ansatz entsprechend gespeichert werden (1). Die Wissensbasis dient als Grundlage für den gesamten Ablauf zur Wiederverwendung des jeweiligen Ansatzes (2). Innerhalb dieser Wissensbasis erfolgt die Suche nach Prozesslösungen. Von den gefundenen Lösungen werden die zur Wiederverwendung am besten geeigneten Artefakte, die die Prozesslösungen repräsentieren ausgewählt und die verschiedenen Strategien der Ansätze

zur reaktiven Wiederverwendung auf die gefundenen Prozesslösungen angewendet, um neue Prozesslösungen zu generieren. Anschließend wird die Konsistenz der neu erstellten Prozesslösungen überprüft und im letzten Schritt die neu geschaffenen Prozesslösungen in der Wissensbasis gespeichert.

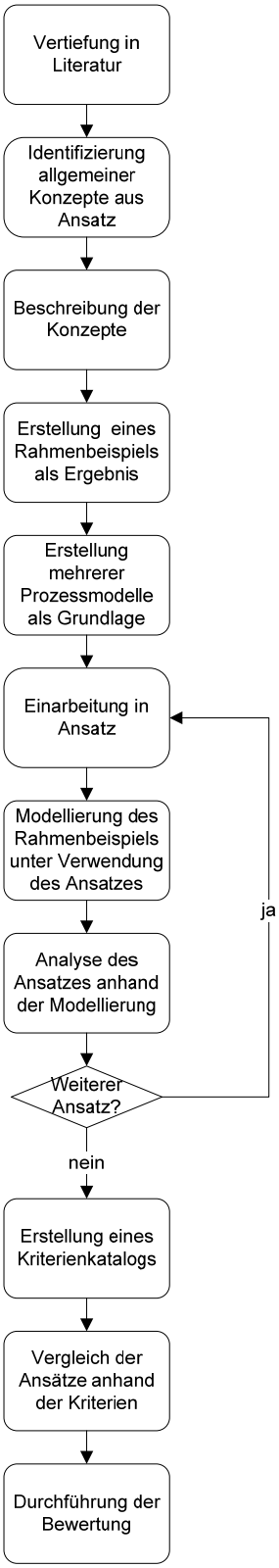



**Abbildung 1: Vorgehensmodell zur Analyse und zum Vergleich der reaktiven Modellierungsansätze**

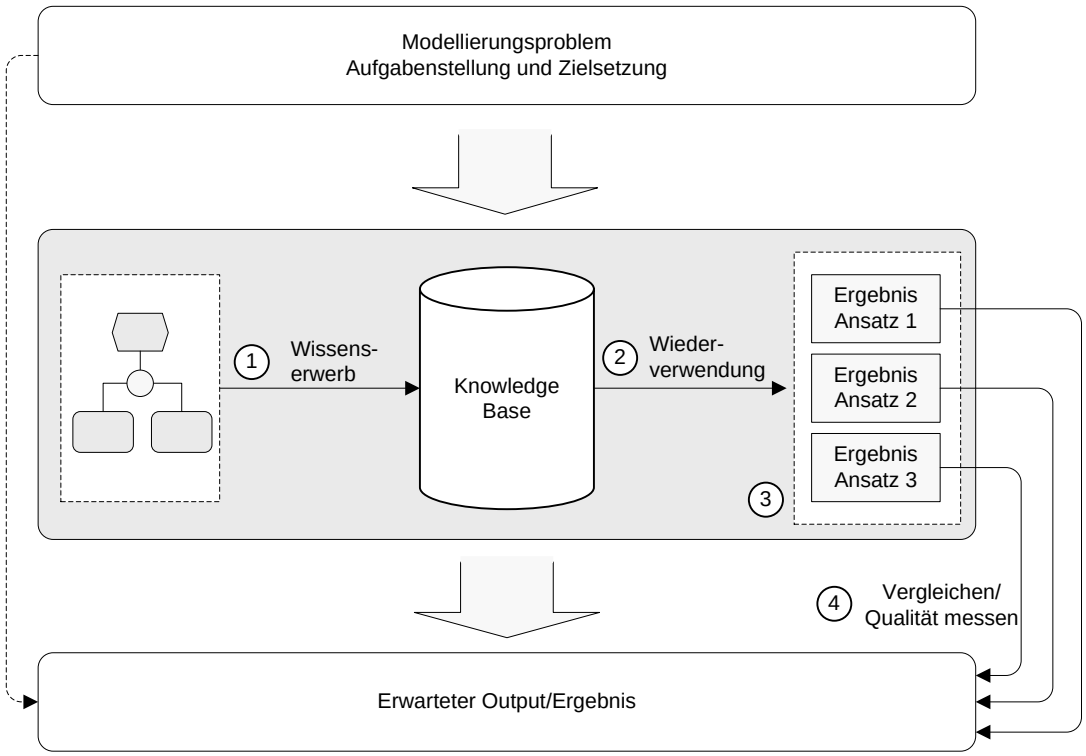



**Abbildung 2: Untersuchungsdesign**

Nach dem gesamten Ablauf wird ein Ergebnis erreicht (3), das mit dem erwarteten Output verglichen wird. Mit anderen Worten, die durch diesen Ablauf generierte Prozesslösung wird mit einem definierten Soll-Output verglichen und gemäß den Anforderungen an die reaktive Wiederverwendung bewertet.

## *1.3 Rahmenbeispiel für den Vergleich der Ansätze*

Die Untersuchung der Ansätze erfolgt anhand mehrerer Beispielprozesse, die als erweiterte ereignisgesteuerte Prozessketten (eEPK) modelliert werden. In Abschnitt 1.3.1 wird die Aufgabenstellung, die die Problemsituation darstellen soll, natürlichsprachig definiert. Der Soll-Output, der die Lösung der Aufgabenstellung darstellt, ist Abschnitt 1.3.2 dargestellt. Eine kurze Einführung in die Modellierungssprache eEPK wird in Abschnitt 2.1.4 gegeben.

### 1.3.1 Aufgabenstellung des Rahmenbeispiels

Als Rahmenbeispiel dient die Bearbeitung eines Hypothekantrags in einer Bank. Die Aufgabe der Bearbeitung eines Hypothekantrags kann in folgende Teilaufgaben zerlegt werden:

- Der Hypothekantrag wird bezüglich der Risikohöhe geprüft, d.h. es wird die Sicherheit bzw. Kreditwürdigkeit des Antragstellers beurteilt. Dazu wird ein Grundbuchauszug, Bewertungsgutachten und Einkommensnachweis benötigt.
- Die maximale Kredithöhe wird ermittelt, wozu zusätzlich eine Insolvenzauskunft benötigt wird.

- Der Hypothekantrag wird bewilligt, wenn das Risiko niedrig ist, oder abgelehnt, wenn es zu hoch ist. Bei mittlerem Risiko soll der Antrag durch den Vorstand nochmals geprüft werden.
- Der Kunde wird über die Entscheidung durch einen Brief informiert.

### 1.3.2 Soll-Output

Das Startereignis des Beispielprozesses in Abbildung 3 ist „Hypothekarkredit ist beantragt“. Dieses Startereignis stößt den Prozesswegweiser „Beleihungsgrenze“ an. Der nachgelagerte Prozess ist in Abbildung 4 dargestellt. Der Prozesswegweiser erzeugt das Ereignis „Kredithöhe ist ermittelt“ als Ergebnis. Dieses löst nun wiederum den Prozesswegweiser „Hypothekarkredit prüfen“ aus (dargestellt in Abbildung 5). Mit dem Endereignis „Hypothekarkredit ist bearbeitet“ ist der Beispielprozess abgeschlossen.

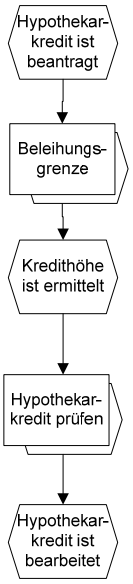



**Abbildung 3: Beispielprozess „Hypothekarkredit bearbeiten“**

Abbildung 4 zeigt den verfeinerten Prozess „Beleihungsgrenze“ aus Abbildung 3. Mit diesem Beispiel wird gezeigt, wie eine bessere Übersichtlichkeit geschaffen werden kann. Dazu wird wieder das Ereignis „Hypothekarkredit ist beantragt“ als Startereignis verwendet. Mit der UND-Verknüpfung wird das Beschaffen folgender Unterlagen parallelisiert: „Grundbuchauszug beschaffen“, „Bewertungsgutachten beschaffen“, „Einkommensnachweis beschaffen“ und „Insolvenzauskunft beschaffen“. Nachdem der Grundbuchauszug und das Bewertungsgutachten verfügbar sind (Ereignisse „Grundbuchauszug verfügbar“ und „Bewertungsgutachten verfügbar“), führt eine UND-Verknüpfung diese Ereignisse zusammen und es folgt die Funktion „Beleihungsgrenze ermitteln“. Dasselbe geschieht bei der Funktion „Pfändbares Gehalt ermitteln“ (Zusammenführen von „Einkommensnachweis verfügbar“ und „Insolvenzauskunft verfügbar“). Die Ereignisse werden anschließend wieder mit dem UND-Konnektor zusammengeführt, und es folgt die Funktion „Max. Kredithöhe berechnen“. Mit dem Endereignis „Kredithöhe ist ermittelt“ ist der durch den Prozesswegweiser nachgelagerte Beispielprozess beendet.

Abbildung 5 zeigt den verfeinerten Prozess „Hypothekarkredit prüfen“ aus Abbildung 3. Das Startereignis dieses Prozesspfads ist das Ereignis „Kredithöhe ist ermittelt“. Dieses löst die

Funktion „Kreditrisiko/Bonität prüfen“ aus. Der nachfolgende Verknüpfungsoperator XOR bedeutet ein exklusives ODER. Dies signalisiert, dass nur einer der drei Pfade verfolgt werden darf. Ist das „Kreditrisiko mittel“, muss das Kreditrisiko nochmals überprüft werden, diesmal durch den Vorstand. Die drei weiteren XOR-Verknüpfungen bedeuten eine Zusammenführung dieser Verzweigungen. Ist das „Kreditrisiko niedrig“ wird die Funktion „Kreditantrag bewilligen“ ausgelöst. Ist das „Kreditrisiko hoch“ wird die Funktion „Kreditantrag ablehnen“ ausgelöst. Der darauf folgende Verknüpfungsoperator (XOR-Verknüpfung) führt wiederum diese Verzweigungen zusammen. Mit der UND-Verknüpfung wird eine Parallelisierung der Funktion „Kunde informieren“ und des Prozesspfades „Kreditvertrag erstellen“ nach dem Ereignis „Kreditantrag bewilligt“ dargestellt. Durch das Endereignis „Hypothekarkredit ist bearbeitet“ ist dieser Prozess beendet. Auf eine Modellierung des Prozesspfades „Kreditvertrag erstellen“ wird verzichtet, da dies keinen Mehrwert für die Untersuchung liefert und daher das Ergebnis nicht beeinflusst.

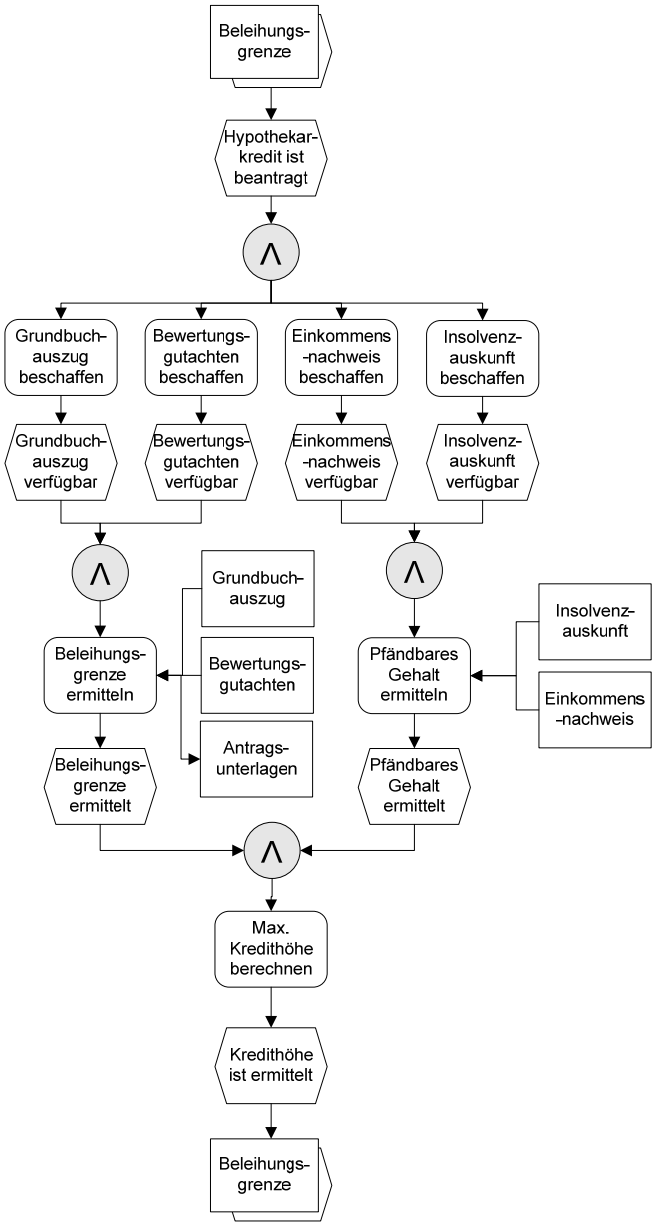



**Abbildung 4: Darstellung von „Beleihungsgrenze“**

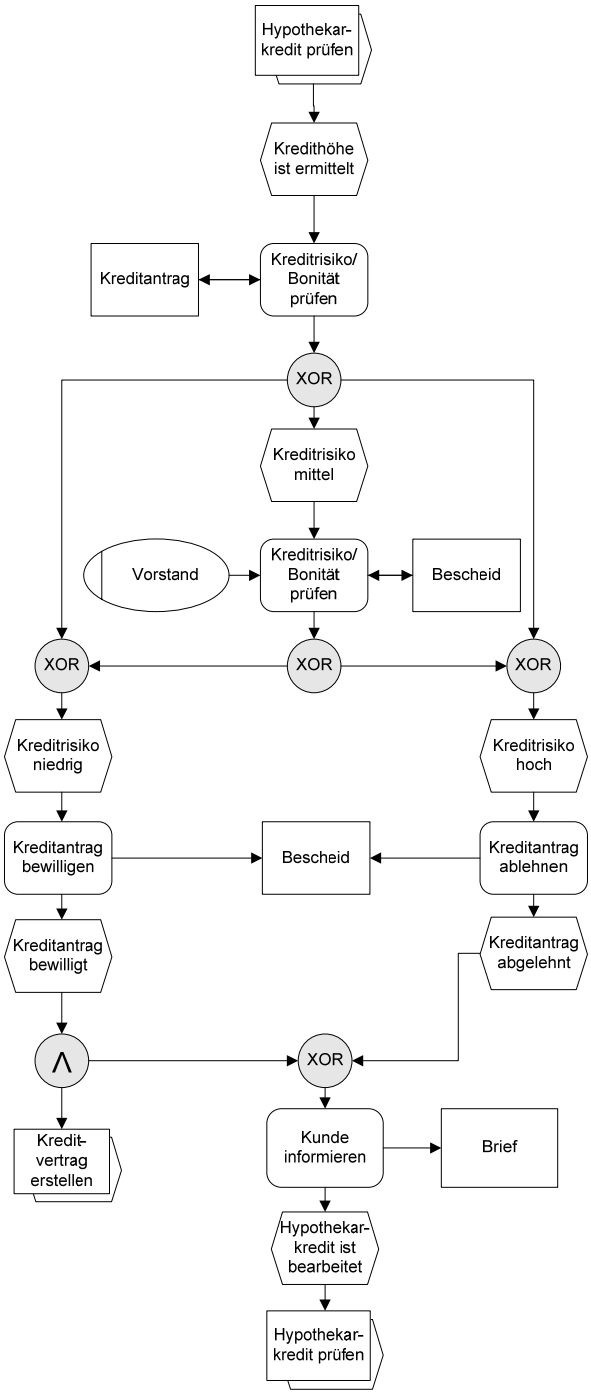



**Abbildung 5: Darstellung von „Hypothekarkredit prüfen“**

## *1.4 Prozessmodelle*

Die in diesem Abschnitt dargestellten Prozessmodelle bilden die Grundlage für die Erstellung der Wissensbasis. Sie stellen vorgefertigte Prozesslösungen zur Verfügung, die durch den Prozess der Wiederverwendung durch die jeweiligen Konzepte wiederverwendet werden können.

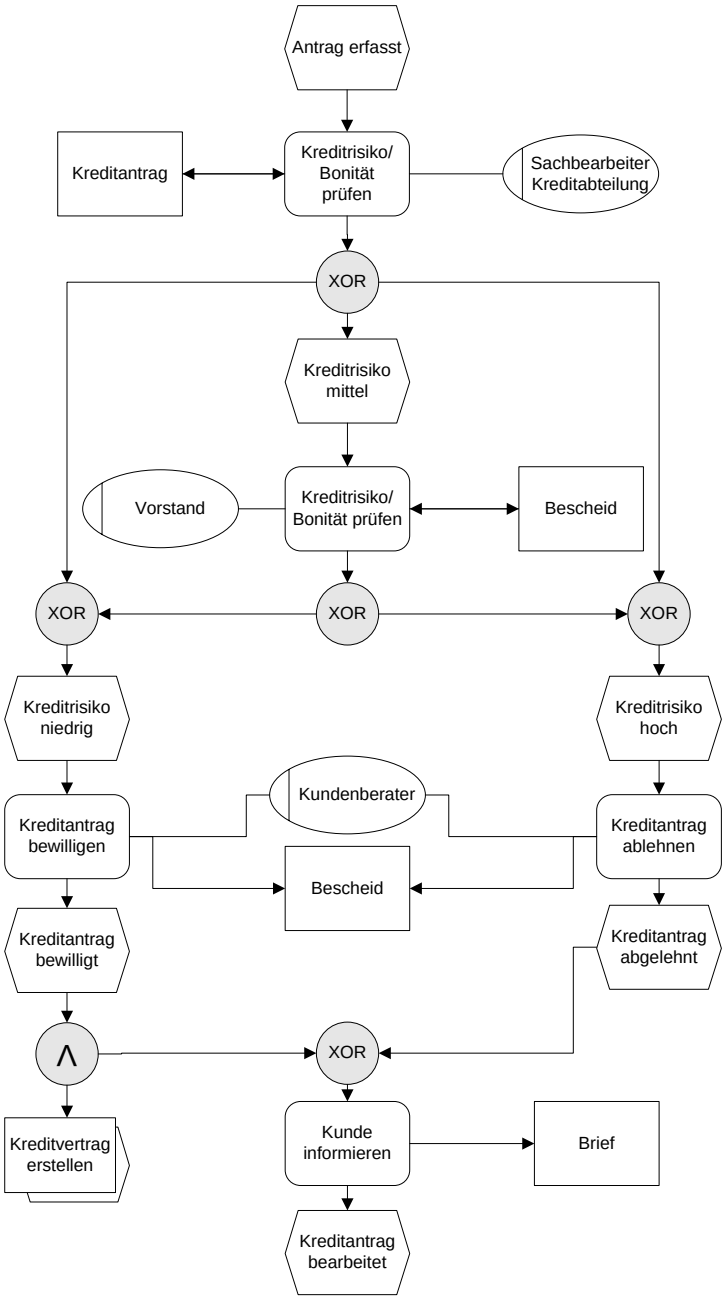



**Abbildung 6: Kreditvergabe mit Bonitätsprüfung, Risiko hoch/mittel/niedrig (model_100)**

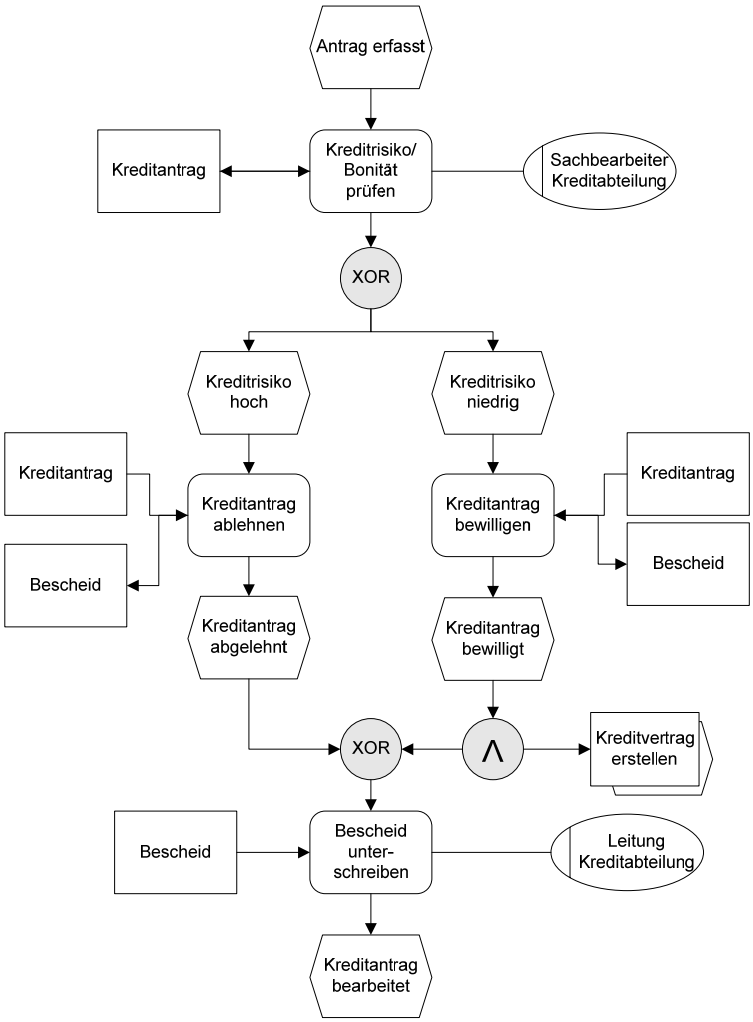



**Abbildung 7: Kreditvergabe mit Bonitätsprüfung, Risiko niedrig/hoch (model_101)**

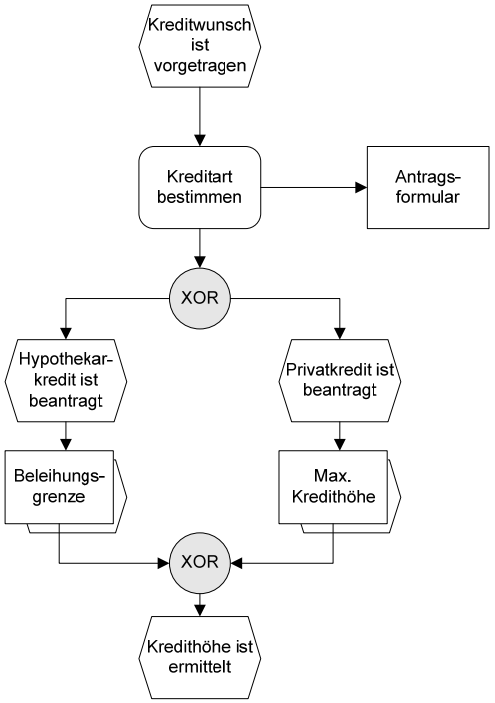



**Abbildung 8: Kredithöhe für Hypothekar- und Privatkredite ermitteln (model_102)**

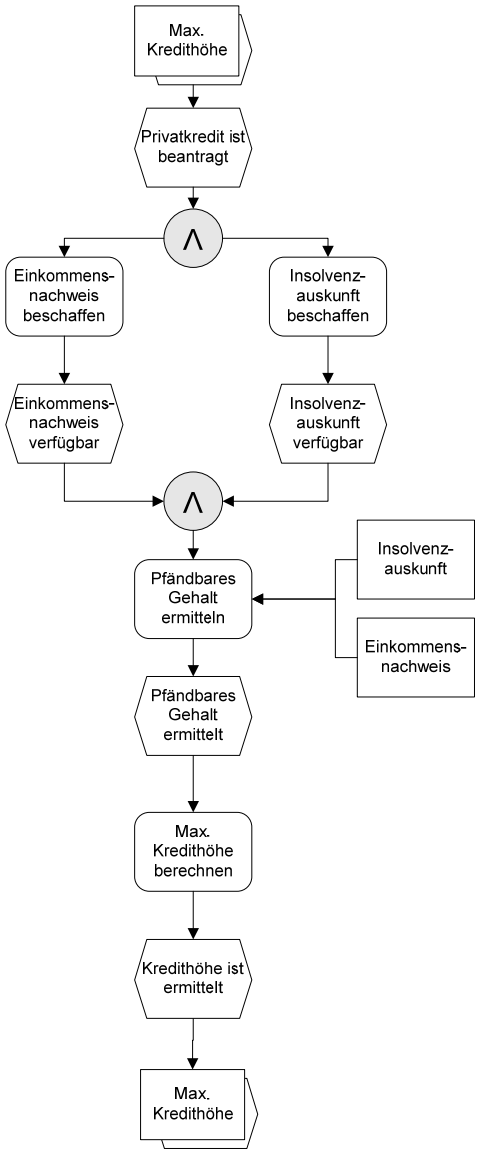



**Abbildung 9: Maximale Kredithöhe ermitteln für Privatkredit (model_1021)**

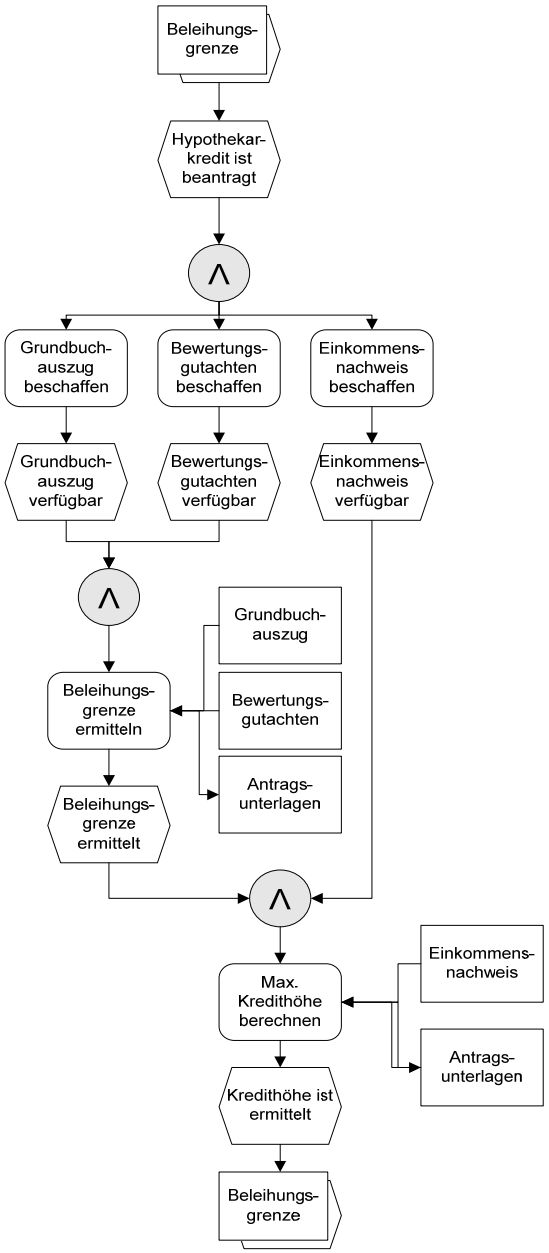



**Abbildung 10: Beleihungsgrenze ermitteln für Hypothekarkredit (model_1022)**

## *1.5 Aufbau dieser Arbeit*

In Kapitel 2 werden die für das Verständnis benötigten Grundlagen erläutert. Dazu erfolgt eine allgemeine Einführung in die Prozessmodellierung und in unterschiedliche Ansätze zur proaktiven und reaktiven Wiederverwendung.

In Kapitel 3 wird der musterbasierte Ansatz nach Hagen, in Kapitel 4 der bausteinbasierte Ansatz nach Rupprecht und in Kapitel 5 nach Krampe anhand gemeinsam identifizierter Konzepte beschrieben. Die von jedem Ansatz zur Verfügung gestellten Strategien zur Wiederverwendung und Anpassung werden durch das eingeführte Rahmenbeispiel veranschaulicht.

Kapitel 6 beschäftigt sich mit der Analyse und dem Vergleich der untersuchten Ansätze anhand aus der Literatur abgeleiteter Anforderungen.

Die Ergebnisse und Schlussbetrachtungen werden anschließend in Kapitel 7 zusammengefasst.

# 2 Grundlagen

In diesem Abschnitt wird zuerst erläutert, was unter Prozessmodellierung zu verstehen ist (Abschnitt 2.1). Anschließend wird eine kurze Einführung in die Wiederverwendung von Prozessmodellen gegeben (Abschnitt 2.2).

## *2.1 Prozessmodellierung*

Um auf die Prozessmodellierung eingehen zu können, muss erst der Begriff *Prozess* erklärt werden (Abschnitt 2.1.1). Nachdem in Abschnitt 2.1.2 auf unterschiedliche Sichtweisen des Modellbegriffs eingegangen wird, werden in Abschnitt 2.1.3 ausgewählte Einsatzbereiche vorgestellt. In Abschnitt 2.1.4 wird kurz auf verschiedene Prozessmodellierungsmethoden eingegangen und die für die Prozessmodelle verwendete Prozessmodellierungsmethode eEPK näher erklärt.

### 2.1.1 Prozess

Es gibt eine Vielzahl an unterschiedlichen Definitionen eines Prozesses in der vorhandenen Literatur [Hess96, 9], wobei hier auf ausgesuchte Autoren eingegangen wird.

Rupprecht definiert einen Prozess als *„eine zusammengehörende Abfolge von Aktivitäten zur Erreichung eines Ziels unter Beteiligung von Ressourcen“*, wobei er eine Aufgabe unterschiedlich zum Prozess sieht [Rupp02, 19]. Laut [Hess96, 20] können Aktivitäten als Teilschritte eines Prozesses gesehen werden. Diese können wiederum zerlegt und in eine sequentielle Abfolge gebracht werden. Die Abfolge der Aktivitäten hängen zeitlich oder logisch zusammen, wobei zeitliche Abfolgen einen Start- und Endzeitpunkt haben und logische Abfolgen unabhängig der zeitlichen Abfolge zu sehen sind [Rupp02, 19]. Ressourcen können Menschen, Methoden, Betriebsmittel oder andere materielle oder immaterielle Sachen[1] sein [Lang97, 10]. Damit ein Prozess angestoßen bzw. ausgelöst werden kann, benötigt er einen Input und das Ziel oder die Leistung des Prozesses ist der Output. Die Aufgabenträger eines Prozesses werden auch Agenten genannt und können Menschen oder aber Maschinen sein [Rupp02, 19f].

[Kram96, 21] sieht einen Prozess als Geschäftsfunktion, analog zu den Geschäftsprozessen. Er betont auch die Prozesssicht, die sich aus Funktionssicht und Kommunikationssicht zusammensetzt. Die Funktionssicht gibt die Geschäftsfunktionen, also Geschäftsprozesse, ihre Organisationseinheiten und die auslösenden Ereignisse wieder. Die Kommunikationssicht zeigt den Datenfluss zwischen den Prozessen und den Agenten, die in dieser Definition ebenfalls Menschen oder Systeme sein können.

[HeHR04, 282f] bezeichnet einen Geschäftsprozess als *„eine Menge von messbaren Tätigkeiten, die für die Schaffung eines spezifischen Ergebnisses für einen bestimmten Kunden oder Markt durchgeführt werden“*. Analog zu der Prozessdefinition nach [Rupp02, 19] können auch Geschäftsprozesse in mehrere Teilprozesse zerlegt werden.

[1] Darunter fallen z.B. Informationen, Werkstoffe, Finanzmittel [Lang97, 10].

Häufig werden die Begriffe *Geschäftsprozess* und *Workflow* synonym verwendet. Dennoch gibt es Unterschiede in der Zielsetzung. Jedoch ist im Gegenzug dazu eine Abgrenzung nicht einfach durchzuführen, da es auch Überlappungen gibt. Die *„Geschäftsprozessmodellierung zielt primär auf die Analyse und Gestaltung betrieblicher Systeme und ihrer Teilsysteme“* und damit auf die Analyse und Optimierung der Geschäftsprozesse. Im Gegensatz dazu konzentriert sich die *Workflowmodellierung* auf *„die Ausführung von Prozessen“* und wird im operativen Bereich eingesetzt. [BeRo96, 50] Einige Autoren sehen Workflows als verfeinerten Geschäftsprozess (vgl. [GaSc95], [Sche98a, b]).

Dies sind nur einige ausgewählte Definitionen, jedoch haben alle gemeinsam, dass Abläufe in Unternehmen in eine zeitlich-logische Abfolge gebracht werden, und durch Aktivitäten oder Tätigkeiten ein bestimmtes, gesetztes Ziel mit den notwendigen Ressourcen erreicht werden soll.

### 2.1.2 Prozessmodell

Der Begriff Modell kann zwischen einem abbildungsorientierten und konstruktionsorientierten Modellbegriff nach Stachowiak unterschieden [Broc03, 9]. Bei dem *abbildungsorientierten* Modellbegriff steht die Abbildung der Wirklichkeit als Modell im Vordergrund [Broc03, 10ff]: Dabei wird von der realen Welt ein Ausschnitt (Diskurs) zuerst durch Interpretation in ein Objektsystem übertragen, das zuerst noch beim Modellierer selbst implizit existiert. Anschließend wird das Objektsystem durch – am besten formale – Sprachkonstrukte in ein Modellsystem überführt. Diese Sichtweise birgt jedoch den Nachteil der Vernachlässigung der *„Subjektivität der Wahrnehmung“* in sich. Eine Verständigung zwischen Subjekten wird erschwert und somit die intersubjektive Modellqualität beeinträchtigt, da nicht angenommen werden kann, dass alle Elemente der Realwelt von unterschiedlichen Subjekten gleich wahrgenommen und somit gleiche Voraussetzungen für ein Modellsystem geschaffen werden können [ScBe98, 82]. Aus diesem Grund ist diese Sichtweise des Modellbegriffs für die *„Gestaltung von Konstruktionsprozessen“* begrenzt geeignet [Broc03, 12]. Dem wirkt die *konstruktionsorientierte* Sichtweise entgegen, da hier die Sichtweise unterschiedlicher Modellierer und Modellnutzer berücksichtigt wird (vgl. z.B. [Luhm99, 181ff], [Schü98, 40ff]). [Broc03] und [Schü98] befassen sich mit den Unterschieden dieser beiden Sichtweisen noch detaillierter, daher wird für eine tiefergehende Lektüre auf diese beiden Werke verwiesen.

Ein Prozessmodell stellt somit entweder eine abbildungsorientierte oder konstruktionsorientierte Sichtweise auf Unternehmen dar. Im Folgenden wird von dem konstruktionsorientierten Begriff ausgegangen, da im Sinne der Wiederverwendung und der damit verbundenen Steigerung der Effizienz und Effektivität der Geschäftsprozesse im Unternehmen eine Abbildung der Wirklichkeit nicht als sinnvoll erachtet werden kann. Nachfolgend wird der Begriff Prozessmodellierung aussagekräftig von [BHKS00] formuliert:

*„Dem konstruktionsorientierten Modellbegriff folgend wird hier ein Informationsmodell definiert als immaterielle Repräsentation eines Objektsystems für Zwecke der Organisations- und Anwendungssystemgestaltung. Das Informationsmodell ist Ergebnis einer Konstruktion eines Modellierers, der Informationen über zu modellierende Elemente eines Systems zu einer Zeit als relevant mit Hilfe einer Sprache deklariert.“*

Dieses Zitat soll durch Ausführungen in Abschnitt 2.1.3 verdeutlicht werden.

### 2.1.3 Einsatzzwecke

Prozessmodelle sollen im gesamten Unternehmensbereich eingesetzt werden, wobei [BeKR05, 51ff] mögliche Einsatzbereiche in *Organisations- und Anwendungssystemgestaltung* unterteilt (vgl. Abbildung 11):

- *Organisationsgestaltung:* Wesentliche Einsatzbereiche der Organisationsgestaltung sind die *Organisationsdokumentation*, *prozessorientierte Reorganisation* (vgl. Business Reengineering weiter unten) und *kontinuierliches Prozessmanagement*, durch das die Prozesse kontinuierlich geplant, überwacht und gesteuert werden (Prozesscontrolling). Unter Organisationsdokumentation wird hier nicht nur die Dokumentation des Aufbaus des Unternehmens (Organigramme), sondern auch die Dokumentation des Ablaufs im Unternehmen verstanden und damit die Beschreibung der Geschäftsprozesse.

  Die Dokumentationen von Prozessen können in weiterer Folge auch für Zwecke der *Zertifizierung nach DIN ISO 9000ff.*, *Benchmarking* sowie *Wissensmanagement* verwendet werden. Eine erfolgreiche Zertifizierung ist laut *„Expertenmeinung zu 50-80 Prozent auf eine qualitativ hochwertige Dokumentation“* zurückzuführen. Der Vergleich von Kennzahlen sowie Unternehmensstrukturen und Geschäftsprozessen mit *Best-Practice*, *Better-Practice* oder auch *Common-Practice* können durch das explizit Machen von Prozesswissen in Form von Prozessmodellen wesentlich unterstützt werden. Ein Vergleich von beispielsweise Kennzahlen wie Prozesskosten und Prozessdurchlaufzeit an Referenzpunkten wird dadurch erleichtert (z.B. mit Hilfe von attributierbaren Prozessmodellen). Ein weiterer Einsatzbereich ist das Wissensmanagement, da es sich bei den Prozessmodellen um explizites Prozesswissen handelt. Das Wissensmanagement beschäftigt sich mit der Identifikation, Akquisition, Nutzung, Weiterentwicklung und Verteilung von Wissen.

- *Anwendungssystemgestaltung:* Der Anwendungssystemgestaltung können beispielsweise wesentliche Einsatzbereiche für Prozessmodelle wie *Auswahl von ERP-Software, modellbasiertes Customizing, Softwareentwicklung, Workflowmanagement* sowie *Simulation* zugeordnet werden. Da sowohl ERP-Software als auch modellbasiertes Customizing durch Konfiguration der Software gekennzeichnet ist, können einerseits die gestellten Anforderungen an die ERP-Software wesentlich einfacher mit den gelieferten Funktionen verglichen werden, da häufig die Dokumentation der Funktionalitäten durch Referenzprozessmodelle erfolgt, die mit den unternehmensspezifischen Prozessmodellen verglichen werden können. Beim Modellbasierten Customizing werden Referenzmodelle durch Parametrisierung der Software dem Unternehmen entsprechend konfiguriert. Sind bereits Prozessmodelle vorhanden, *„wird der Software-Einführungsprozess beschleunigt und qualitativ hochwertiger, da Domänenwissen ohne zusätzliches technisches Wissen für die Softwareeinführung – in weiten Bereichen – ausreichend ist“*. In der Softwareentwicklung werden durch das Requirements Engineering an die zu entwickelnde Software gestellten Anforderungen mittels CASE-Tools modelliert und anschließend umgesetzt. Sogenannte I-CASE-Lösungen (Integrierte CASE-Lösungen) generieren aus den erstellten Modellen Code und führen somit zu Zeit- und Kostenersparnis bei der Softwareentwicklung.

Mit dieser Unterteilung wird das Zusammenspiel zwischen Organisationsgestaltung und Ablauf eines Unternehmens und diese unterstützende Informationssysteme verdeutlicht.

Immer öfter werden Geschäftsprozesse durch Informationssysteme erst ermöglicht. Nach [Öste95, 20f] ist die Prozessentwicklung und damit auch die Erstellung von Prozessmodellen das Bindeglied zwischen Strategieentwicklung und Informationssystementwicklung. In Unternehmen ist eine Vielzahl an Prozessen beteiligt, die Unternehmensstrategie zu verwirklichen.

Da Prozesse auf Grund der sich ständig ändernden Anforderungen der Umwelt als dynamisch anzusehen sind, entstand Forschungsbedarf in der Prozessmodellierung und der Verbesserung und Restrukturierung in Unternehmen. Änderungsbedarf kann durch unterschiedliche Zielsetzungen in der Prozessmodellierung entstehen. [Rose02, 15f] listet dabei Ziele auf wie Ausrichtung der Geschäftsprozessmodellierung auf Kundennutzen und -bedürfnisse und somit der Steigerung der Kundenzufriedenheit; Speicherung des gesamten *„Organisationswissens in Datenbanken (Process Warehouse)“*, um damit unter anderem die Prozesstransparenz zu erhöhen und somit Leistungs- und Kostennachweise für die einzelnen Prozesse zu erhalten.

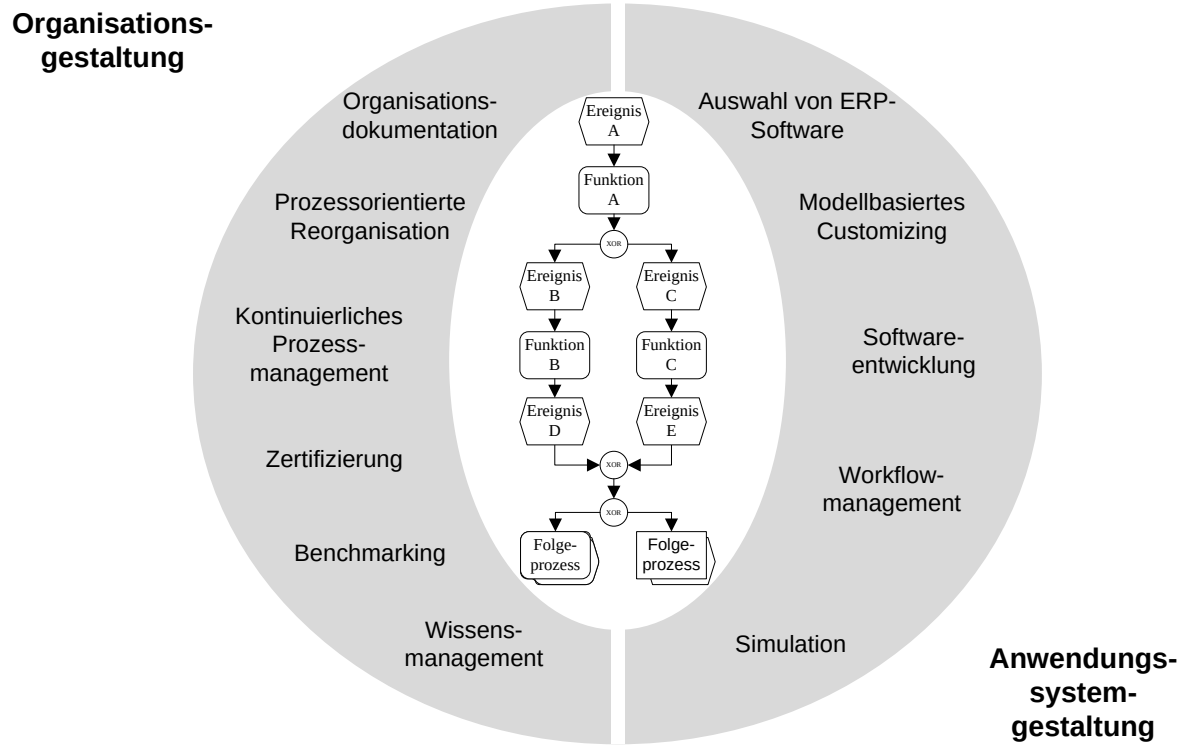



**Abbildung 11: Einsatzzwecke von Prozessmodellen [BeKR05, 57]**

Verschiedene Konzepte wie etwa Business Process Reengineering (vgl. prozessorientierte Reorganisation in Abbildung 11) und Geschäftsprozessoptimierung sind aus den Anforderungen entstanden, Prozesse ständig an sich ändernde Umweltbedingungen anzupassen. Die Begriffe Business Process Reengineering, Business Engineering, Geschäftsoptimierung und Business Redesign werden teilweise synonym verwendet [Gada05, 4ff]. Im Folgenden wird kurz auf die Unterschiede zwischen Business Reengineering und Geschäftsprozessoptimierung eingegangen:

- ***Business Reengineering:*** In den Werken [Hamm90] und [HaCh94] wurde das Konzept des Business Reengineerings veröffentlicht, das von Hammer und Champy selbst als *fundamental*, *radikal* und *dramatisch* bezeichnet wird. Die Idee dieses Konzepts verfolgt eine Umstrukturierung der Unternehmen, und zwar weg von funktionalen Organisationen wie etwa Beschaffung, Produktion und Logistik als

Funktionen hin zu Prozessdenken, bei dem die erstellten Leistungen in den Vordergrund rücken (z.B. Vertriebsprozess, Beschaffungsprozess etc.). Diese Prozesse sind als Tätigkeiten zu verstehen, die funktionenübergreifend durchgeführt werden. Dabei wird der Fokus auf die Hauptprozesse wie Ausrichtungen auf Kundenbedürfnisse und -nutzen (z.B. Vertriebs-/Produktionsprozesse) gelegt, die durch Nebenprozesse unterstützt werden [Gada05, 4].

Das Konzept Business Reengineering wurde intensiv weiterentwickelt und *„im deutschsprachigen Raum“* konnten sich die Ansätze nach Scheer und Österle etablieren [Gada05, 8]. Dabei werden immer öfter Informationssysteme als Enabler für den wirtschaftlichen Erfolg angesehen. [Sche97] hat durch ARIS (Architektur integrierter Informationssysteme) sowohl ein Konzept als auch eine Softwarelösung geschaffen, mit dessen Hilfe Prozesse des gesamten Unternehmens modelliert, analysiert und auch optimiert werden können. Zur Komplexitätsbeherrschung unterteilt er das Unternehmen in verschiedene Sichten (Datensicht, Funktionssicht, Organisationssicht und Ressourcensicht) und Ebenen (Fachkonzept, DV-Konzept und technische Implementierung). Abbildung 12 zeigt eine Übersicht über die Sichten und Ebenen. Für eine detailliertere Ausführung sei hier auf Scheer verwiesen. [Öste95, 14ff] sieht das Business Reengineering als Top-Down-Ansatz und definiert dabei folgende drei Ebenen: Geschäftsstrategie, Prozess und Informationssystem. Ausgehend von der Entwicklung einer Geschäftsstrategie werden Prozesse definiert und diese mit Hilfe von Informationssystemen umgesetzt und dabei die wesentlichen Dimensionen Organisation, Daten und Funktionen berücksichtigt. Weitere Dimensionen wie Personal sind zwar nicht minder wichtig für Unternehmen, werden jedoch in der Disziplin Wirtschaftsinformatik berücksichtigt (vgl. Abbildung 13).

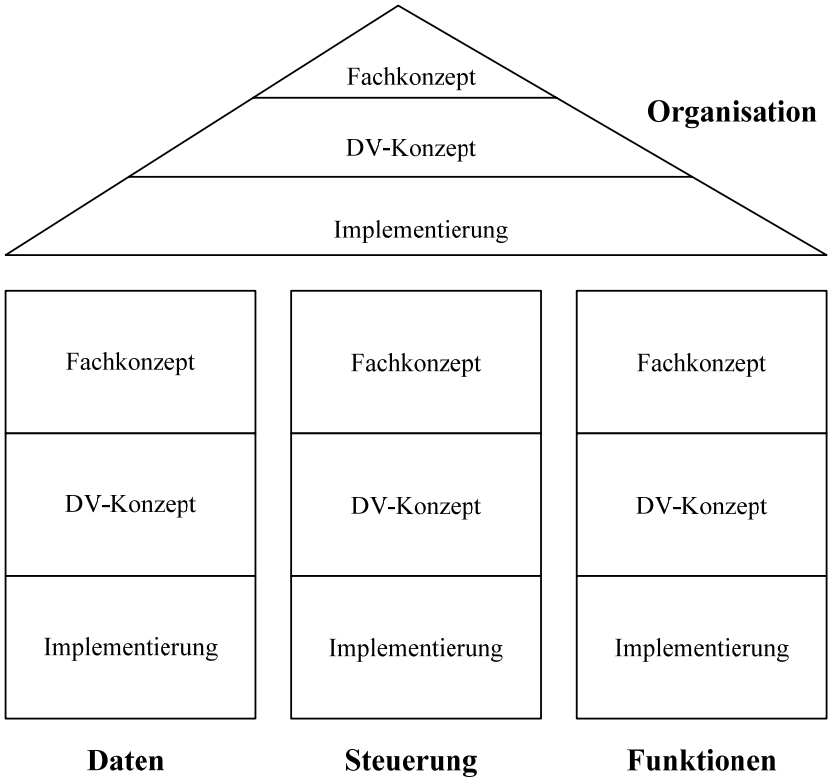



**Abbildung 12: ARIS-Haus in Anlehnung an [Sche96]**

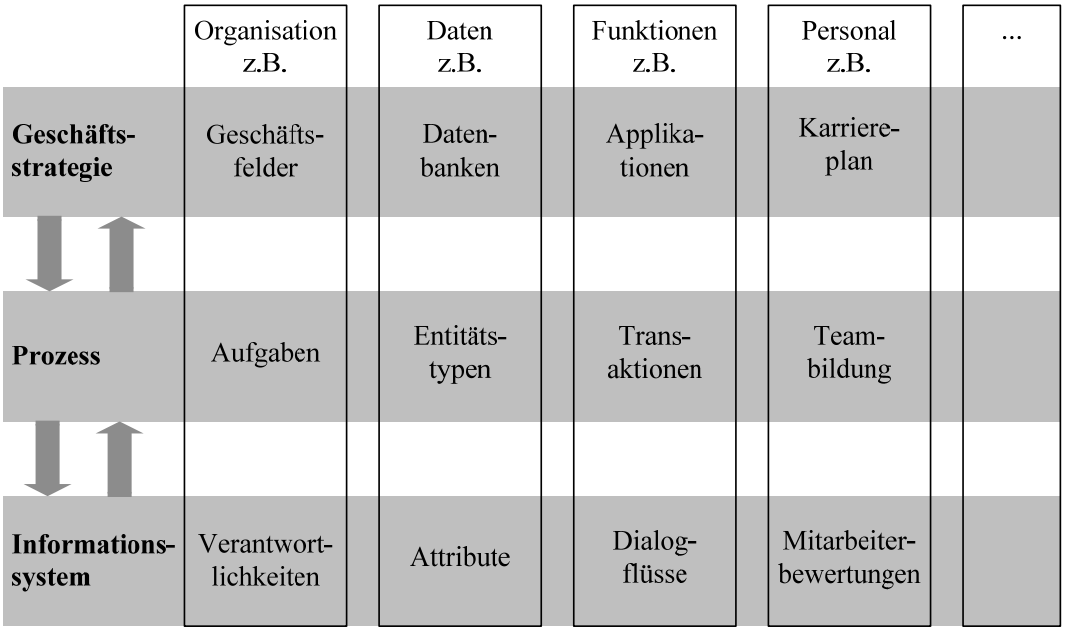



**Abbildung 13: Dimensionen des Business Process Reengineerings [Öste95, 30]**

Durch eine neuerliche Weiterentwicklung des Konzepts des Business Reengineering wurde der Notwendigkeit Rechnung getragen, die Prozesse über die Unternehmensgrenzen hinaus zu betrachten und so Kunden sowie Lieferanten in die Geschäftsprozesse mit einzubeziehen und nicht nur die Prozesse im Unternehmen zu betrachten [Gada05, 34].

- ***Geschäftsprozessoptimierung:*** Nicht ganz so radikal wie das Konzept des Business Process Reengineering ist das Konzept der Geschäftsprozessoptimierung. Obwohl die Begriffe teilweise auch als Synonyme verwendet werden, bezeichnet Geschäftsprozessoptimierung Veränderungen in Unternehmensstrukturen *„in kleineren, aber überschaubaren und weniger riskanten Schritten“*. Als wesentliche Ziele werden hier Verkürzungen von Durchlaufzeiten und Verbesserungen der Prozessqualität genannt. [Gada05, 14ff] Dabei nennt [Blei91, 196] folgende Möglichkeiten, Prozessoptimierungen durchzuführen: Weglassen, Auslagern, Zusammenfassen, Parallelisieren, Verlagern, Beschleunigen und Ergänzen. Beim *Weglassen* wird überprüft, ob alle Aktivitäten notwendig sind, um zu dem gewünschten Ergebnis zu kommen und ob Medienbrüche vermieden werden können. Durch das *Auslagern* können Aktivitäten oder Aufgaben vergeben werden und z.B. durch externe Stellen durchgeführt werden (Outsourcing). Beim *Zusammenfassen* werden Aktivitäten zusammengelegt. Mit der *Parallelisierung* kann die Durchlaufzeit verkürzt werden, indem Aktivitäten parallel durchgeführt werden. Durch *Verlagern* können nachgelagerte Aktivitäten früher durchgeführt werden und somit die Durchlaufzeit ebenfalls verkürzt werden. Durch *Beschleunigen* werden Aktivitäten in einer kürzeren Dauer durchgeführt. Mit der Möglichkeit der *Ergänzung* von zusätzlichen Prozessschritten aber auch Teilprozessen kann die Qualität des Ergebnisses verbessert werden.

Teilweise ergänzend aber auch überschneidend gibt es verwandte Management-Konzepte, *„die zum Teil die Gedanken des Business Reengineering bzw. der Geschäftsprozessoptimierung beinhalten“*. Dazu zählen das Lean Management Konzept, der Kontinuierliche Verbesserungsprozess und SixSigma und werden nachfolgend erklärt: [Gada05, 33f]

- ***Lean Management Konzept:*** Das vom MIT (Massachusetts Institute of Technologie) entwickelte Lean Management Konzept versucht durch *schlanke Unternehmensführung* zu Beginn hauptsächlich in der Produktion und später im gesamten Unternehmen Ziele wie Effizienzerhöhung, Schnelligkeit und bessere Qualität zu erreichen.

- ***Kontinuierlicher Verbesserungsprozess:*** Ein weiteres Konzept hat seinen Ursprung in Japan und ist unter dem Namen Kontinuierlicher Verbesserungsprozess (KVP) bekannt. Dabei liegt der Fokus auf der Verbesserung der Prozesse durch *„Einbindung der Mitarbeiter“*, die durch eingebrachte Vorschläge helfen, Prozesse zu verbessern.

- ***SixSigma:*** Durch das statistische Konzept SixSigma sollen Schwankungen in der Prozessleistung gemessen und einer Null-Fehler-Qualität angepasst werden, da angenommen wird, dass diese Schwankungen die Prozessqualität verschlechtern und damit die Kundenzufriedenheit verringern.

Der Gegenstand des Business Reengineering, der Geschäftsprozessoptimierung und ähnlicher Konzepte ist die Optimierung von Geschäftsprozessen. Es wird versucht, von „Besseren“ bzw. von gemachten Erfahrungen zu lernen, wobei die Wiederverwendung von Prozessmodellen durch Referenzmodellierung zum Einsatz kommt [Gada05, 34], und auch Ansätze der reaktiven Wiederverwendung einen Beitrag zur Optimierung leisten können. Dies wird in Abschnitt 2.2 erklärt.

### 2.1.4 Prozessmodellierungsmethoden

Zur Modellierung von Prozessen gibt es bereits zahlreiche Methoden, die von [Gada05, 66f] in skriptbasierte und grafische Methoden eingeteilt werden. Die skriptbasierten Methoden stellen formale Notationen zur Verfügung, die an Programmiersprachen angelehnt sind. Dies erschwert jedoch die Anschaulichkeit der Prozesse. Durch grafische Methoden werden Prozesse mit Hilfe von Diagrammen dargestellt, die er noch weiter in datenflussorientierte, kontrollflussorientierte und objektorienterte Methoden unterteilt. Bekannter und in der Praxis weit verbreiteter Vertreter der kontrollflussorientierten Methoden ist die ereignisgesteuerte Prozesskette (eEPK). Vertreter der objektorientierten Methoden, die in der Praxis ebenfalls immer mehr an Bedeutung erlangen, sind UML-Aktivitätsdiagramme und UML-Use-Case-Diagramme. Für eine vertiefende Lektüre in die UML Spezifikation sei hier auf [UML2] verwiesen. Im Folgenden wird eine kurze Einführung in die ereignisgesteuerte Prozesskette gegeben.

Eine ereignisgesteuerte Prozesskette (EPK) wird mit Hilfe eines gerichteten Graphen dargestellt [GrMü06, 24], deren Grundelemente Ereignisse und Funktionen sind [LiKo06, 1535]. Funktionen können Aktivitäten der Geschäftsprozesse zugeordnet werden und Ereignisse lösen entweder eine Funktion aus (Input) oder sind das Ergebnis einer Funktion (Output) [LiKo06, 1535]. Durch diese Abfolge können Geschäftsprozesse eines Unternehmens dargestellt werden.

Die EPK wurde zu der eEPK weiterentwickelt. Dabei wurden die Elemente *Stelle*, *Organisationseinheit*, *Informationsobjekt* und *Prozesswegweiser* eingeführt [GrMü06, 33ff]. In Tabelle 1 werden die Elemente der eEPK übersichtlich dargestellt. Ein *Ereignis* ist ein eingetretener Zustand, die eine Funktion auslöst. Eine *Funktion* führt zu einer Änderung des

Zustands und führt zu einem nachfolgenden Ereignis als Ergebnis [BBCD04, 13]. Funktion und Ereignisse können entweder durch eine *Kante* oder durch Verknüpfungsoperatoren (*Konnektoren*) verbunden werden. Eine Verbindung durch Kanten bezeichnet eine sequenzielle Abfolge. Konnektoren können durch eine UND-Verknüpfung eine Parallelität, durch eine exklusive ODER-Verknüpfung (XOR) eine Verzweigung und durch eine ODER-Verknüpfung (OR) eine Disjunktion darstellen [GrMü06, 27]. Eine XOR-Verknüpfung bedeutet, dass nur *ein* nachfolgender Pfad verfolgt werden darf. Bei einer OR-Verknüpfung kann ein Pfad bzw. können mehrere oder alle Pfade verfolgt werden. Dadurch wird der Kontrollfluss gesteuert und somit die zeitlich-logische Abfolge von Ereignissen und Funktionen dargestellt. Eine Funktion kann durch eine *Organisationseinheit* durchgeführt werden, wobei ein oder mehrere *Informationsobjekte* (Daten) konsumiert bzw. erzeugt werden [GrMü06, 34]. Ein *Prozesswegweiser* bedeutet, dass ein weiterer Prozess auf derselben Hierarchie nachgelagert ist [BBCD04, 13]. Dadurch kann die Komplexität eines Prozessmodells reduziert werden, was zu einer besseren Übersichtlichkeit führt. Mit diesem Symbol wird somit ein höheres Abstraktionsniveau erreicht, und der Prozess kann in weiterer Folge in einer separaten eEPK modelliert werden [GeHK99, 385].

Jeder Geschäftsprozess beginnt mit mindestens einem Startereignis und endet mit mindestens einem Endereignis [NüRu02, 69], was schließlich das Ergebnis (Output) eines Geschäftsprozesses ist.

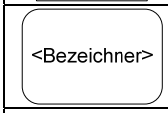
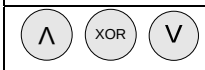
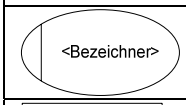
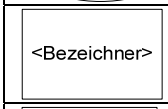
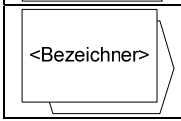
| Elemente | Bedeutung |
|---|---|
|  | Ereignis |
|  | Funktion |
|  | Kante (Verbindung zwischen Ereignis und Funktion) |
|  | Konnektoren UND (Konjunktion), exklusives ODER (Antivalenz) und ODER (Disjunktion) |
|  | Organisationseinheit |
|  | Informationsobjekt |
|  | Prozesswegweiser |

**Tabelle 1: Elemente des eEPK [GrMü06, 24ff]**

Es gibt auch andere Darstellungsweisen (vgl. [KeNS92, 15]). Für die Beispielprozesse wird jedoch die Darstellung nach [GrMü06, 20ff] gewählt. Für eine detailliertere Einführung in die Modellierungssprache EPK sei hier unter anderem auf [KeNS92] verwiesen.

In [Liko06] werden ausgesuchte Prozessmodellierungsmethoden nach Anforderungen evaluiert und die optimalen Einsatzbereiche aufgedeckt: UML Aktivitätsdiagrammme, Business Process Definition Metamodel (BPDM), Business Process Modelling Notation (BPMN), ereignisgesteuerte Prozesskette (EPK), Integrated DEFinition Method 3 (IDEF3), Petri Netze und Role Activity Diagram (RAD).

Zur einheitlichen Darstellung von Daten, die zu Entitäten zusammengefasst werden, und ihren Beziehungen zueinander sind Entity-Relationship-Modelle (ERM) ebenfalls weit verbreitet. Diese Modellierungsmethode wurde bereits in den 70er-Jahren von Chen entwickelt [Chen76].

[Hage05, 28ff] stellt in ihrer Arbeit Prozessmodellierungssprachen (PML) gegenüber, die für die Analyse und Entwicklung von Softwareprozessen verwendet werden können und mehr oder weniger praxistauglich sind. Dabei stellt sie aus unterschiedlichen Paradigmen jeweils zwei Vertreter vor. Zu den Paradigmen zählen hier regelbasierte PML, zustandsbasierte PML, funktionale PML, prozedurale PML und objektorientierte PML. UML und EPKs werden hier objektorientierten PMLs zugeordnet. Auf die weniger praxistauglichen PMLs wird in der vorliegenden Arbeit nicht eingegangen, sondern ist aus [Hage05, 28ff] zu entnehmen.

## *2.2 Wiederverwendung von Prozessmodellen*

Die Ansätze zur Wiederverwendung beschäftigen sich mit der Wiederverwendung von Prozessmodellen. Sie können in Ansätze der proaktiven und reaktiven Wiederverwendung unterteilt werden [BöKS06, 2]. Ansätze der proaktiven Wiederverwendung modellieren die wieder zu verwendenden Artefakte bereits vor der konkreten Anwendung, während in der reaktiven Wiederverwendung die wieder zu verwendenden Artefakte ausgewählt und wiederverwendet werden, wenn der Bedarf danach eintritt. Grundsätzlich können die verschiedenen Ansätze auch kombiniert werden [BöKS06, 14], wobei beispielsweise ein bausteinbasierter Ansatz als Vertreter der reaktiven Wiederverwendung auf Referenzmodellierung zurückgreifen kann, und diese Prozessmodelle in die Wissensbasis integriert werden können.

Auf die Referenzmodellierung als Vertreter der proaktiven Wiederverwendung wird in Abschnitt 2.2.1. Unterschiedliche Vertreter der reaktiven Wiederverwendung werden in Abschnitt 2.2.2 genannt und gemeinsam verwendete Konzepte identifiziert.

### 2.2.1 Proaktive Wiederverwendung

Zu dieser Art von Ansätzen zählt unter anderem die Referenzmodellierung, wobei der Begriff Referenzmodell unterschiedlich betrachtet werden kann (vgl. z.B. [Hars94, 15ff], [Schü98, 37ff]). Grundsätzlich kann ein Referenzmodell als Vorbild zur Ableitung spezifischer Unternehmensmodelle verwendet werden [HeHR04].

Referenzmodelle zeichnen sich durch die Merkmale Allgemeingültigkeit und Empfehlungscharakter aus [Broc03, 31f], um der Forderung nach einem Vorbild gerecht zu werden: Durch das Merkmal *Allgemeingültigkeit* wird angesprochen, dass Referenzmodelle von unternehmensspezifischen Merkmalen abstrahieren sollen und so für eine breitere Masse von Unternehmen einer Anwendungsdomäne wiederverwendbar sind (für eine Erläuterung der Problematik, die durch diese Wortwahl entstehen kann vgl. [Broc03, 31f]). Mit dem Merkmal *Empfehlungscharakter* sollen die Referenzmodelle der Vorbildrolle als Sollmodelle gerecht werden („best practice“). Jedoch weisen sowohl [Schü98, 300] als auch [Broc03, 32] darauf hin, dass das Gehalt der Empfehlung schwierig festzustellen ist und der Empfehlungscharakter kaum zu überprüfen ist. Erst während der Anwendung kommt zu Tage, ob es sich bei den Referenzmodellen tatsächlich um „best practices“ handelt.

Die Forderungen nach Allgemeingültigkeit und Empfehlungscharakter der Referenzmodelle werden von [BDKK02, 26] als Dilemma bezeichnet: Je mehr die Referenzmodelle Allgemeingültigkeit besitzen und somit für einen breiteren Markt zugänglich sind, desto höher ist der Anpassungsaufwand an die spezifischen Unternehmensstrukturen. Sind jedoch in Referenzmodellen bereits erkennbare Unternehmensstrukturen vorhanden, wird zwar der Anpassungsaufwand an das jeweilige Unternehmen verringert, aber dadurch auch die Wiederverwendungsreichweite verringert, da die Referenzmodelle von weniger Unternehmensklassen als Vorlage verwendet werden können.

Referenzmodelle können in nicht-generische und generische Referenzmodelle unterteilt werden [BöKS06, 4]:

- ***Nicht-generische Referenzmodelle:*** Die Anpassung abstrakter Referenzmodelle an spezifische Prozessmodelle wird durch keine Mechanismen zur Anpassung unterstützt. Die Referenzmodelle dienen hier *„als statische Vorlage“* [BöKS06, 4]. Bei dieser Art der Referenzmodellierung besteht jedoch die Gefahr, dass durch fehlende Unterstützung von Anpassungsmechanismen keine Zeit- oder Kostenersparnis gegenüber der Erstellung neuer Prozessmodelle erreicht werden kann. Zur Anpassung dieser Art von Referenzmodellierung wird auf folgende Methodik zurückgegriffen: *„Nicht-generische Referenzmodelle können auf Sprachen zurückgreifen, die bei der Erstellung unternehmensspezifischer Informationsmodelle verwendet werden…, während Sprachen generischer Referenzmodelle um Konstrukte zur Variantenabbildung erweitert werden müssen.“* [BHKS00] Als Beispiel erwähnen die Autoren das Handelsreferenzmodell in [Beck96]. [Rupp02, 2] unterstreicht bei dieser Vorgehensweise die Notwendigkeit einer eventuell hohen Zahl notwendiger Anpassungen.

- ***Generische Referenzmodelle:*** Generische, konfigurierbare oder konfigurative Referenzmodelle werden als Synonyme verwendet (vgl. [BöKS06, 4], [BDKK02]). Konfigurierbare Referenzmodelle können eine Lösung zu dem weiter oben erwähnten Dilemma bieten [BDKK02, 26]. Durch Mechanismen zur Anpassung können hier Referenzmodelle an eine spezifische Anwendungssituation angepasst werden. *„Generische Referenzmodelle enthalten im Gegensatz zu nicht-generischen Referenzmodellen Varianten, die über explizit definierte Anpassungspunkte im Rahmen der Referenzmodellanwendung auswählbar sind.“* [BHKS00]

Da nicht-generische Referenzmodelle nur als Vorlage dienen und keine Mechanismen zur Anpassung an den jeweiligen Anwendungskontext zur Verfügung stellen, wird im Folgenden genauer auf konfigurierbare Referenzmodelle eingegangen.

Die zur Konfiguration notwendigen *Konfigurationsparameter* beschreiben den Anwendungskontext eines Referenzmodells. Sie können in *Unternehmensmerkmale* und *Perspektiven* unterteilt werden [BDKK02, 27ff]:

- ***Unternehmensmerkmale:*** Die Ausprägungen dieser Konfigurationsparameter beschreiben die *Klasse der Unternehmen*. Durch Auswahl solcher Ausprägungen werden durch Regelauswertung *„betriebswirtschaftlich-inhaltliche Gestaltungsempfehlungen des Referenzmodells“* bestimmt. Es kommt dabei je nach Ausprägung zur Verwendung spezifischer Funktionen, spezieller Abläufe etc. Dies bedeutet, dass je nach ausgewählter Geschäftsart die benötigten Funktionsbereiche

ausgewählt werden. Je nach Auswahl können gesamte Funktionsbereiche wegfallen. Bei den Prozesslösungen der Referenzmodelle bedeutet dies, dass entweder einzelne Prozessschritte entfallen oder hinzukommen können (vgl. Abbildung 14).

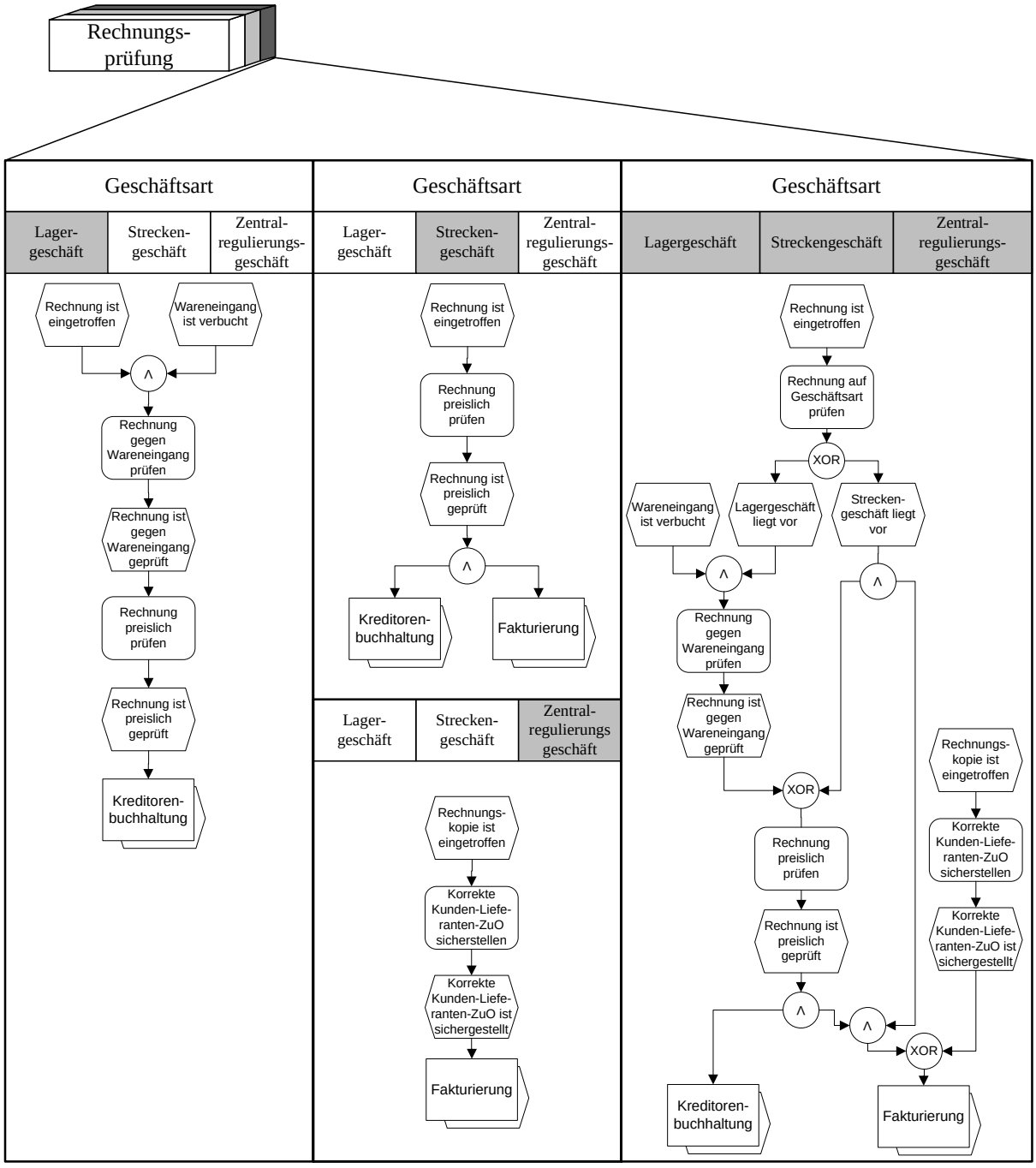



**Abbildung 14: Unternehmensmerkmalskonfiguration feingranular [BDKK02, 28]**

- ***Perspektiven:*** Mit diesem Konfigurationsparameter wird die Sichtweise auf das gesamte Referenzmodell bestimmt. Die Gestaltungsempfehlungen, die durch den Konfigurationsparameter *Unternehmensmerkmal* vorgeschlagen werden, bleiben gleich. Dadurch wird der bereits erwähnten *Subjektivität* Rechnung getragen, die den konstruktionsorientierten Modellbegriff prägt.

  Bei grober Betrachtungsweise werden bei den Perspektiven *Organisationsgestaltung* und *Anwendungssystemgestaltung* für Organisationsgestalter unterschiedliche Modelle

von Bedeutung als für Anwendungssystemgestalter (vgl. Abbildung 15). Den Organisationsgestaltern werden hier Organigramme, EPKs und Fachbegriffsmodelle zugeordnet, während den Anwendungssystemgestaltern Anwendungssystem-architekturen, EPKs und Entity-Relationship-Modelle zugeordnet werden.

Bei feiner Betrachtungsweise werden der Perspektive Organisationsgestaltung Prozessmodelle geliefert, die für diese Perspektive von Bedeutung sind und der Perspektive Anwendungssystemgestaltung Prozessmodelle, die die Anwendungssystemgestaltung betreffen (vgl. Abbildung 16).

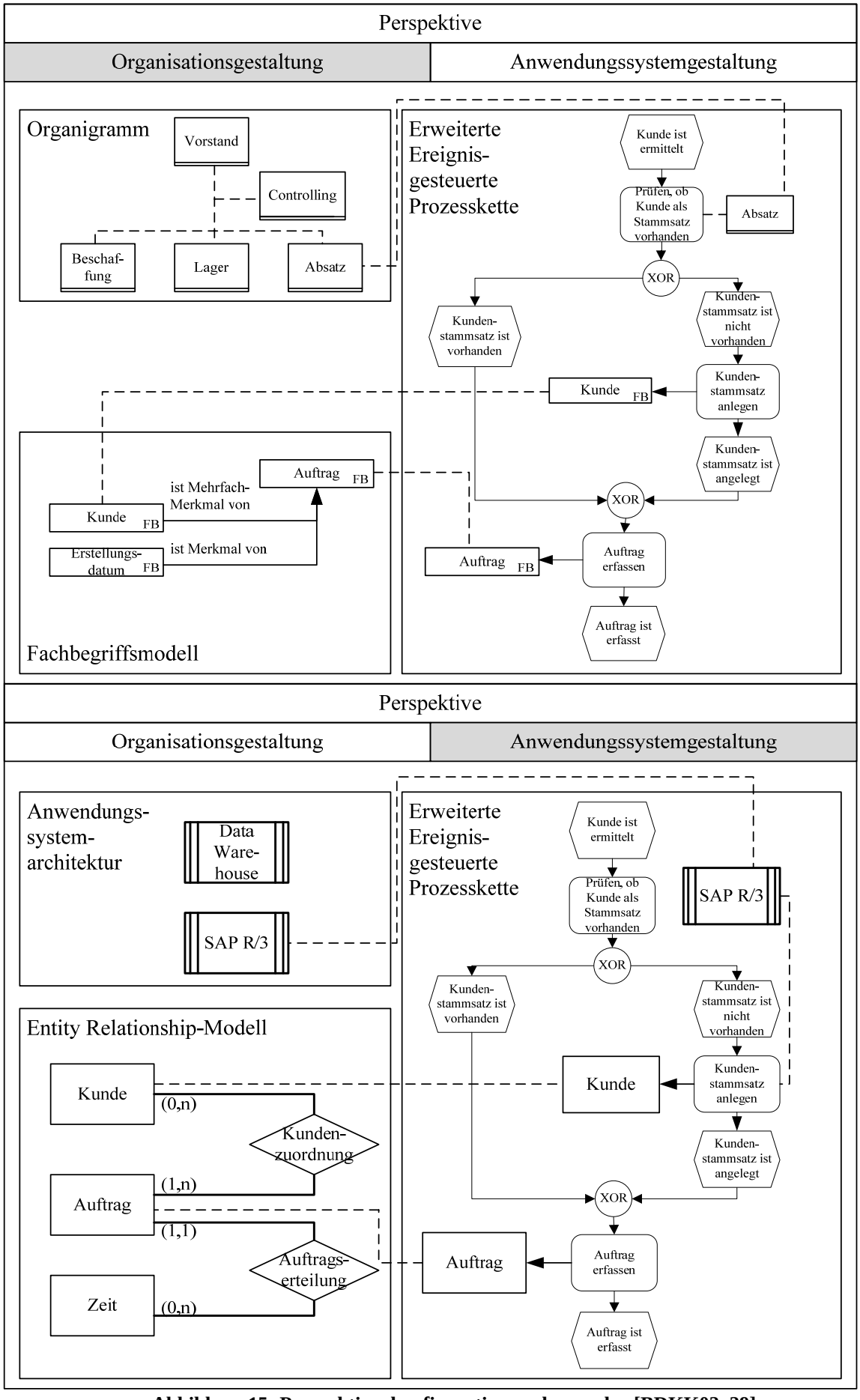



**Abbildung 15: Perspektivenkonfiguration grobgranular [BDKK02, 29]**

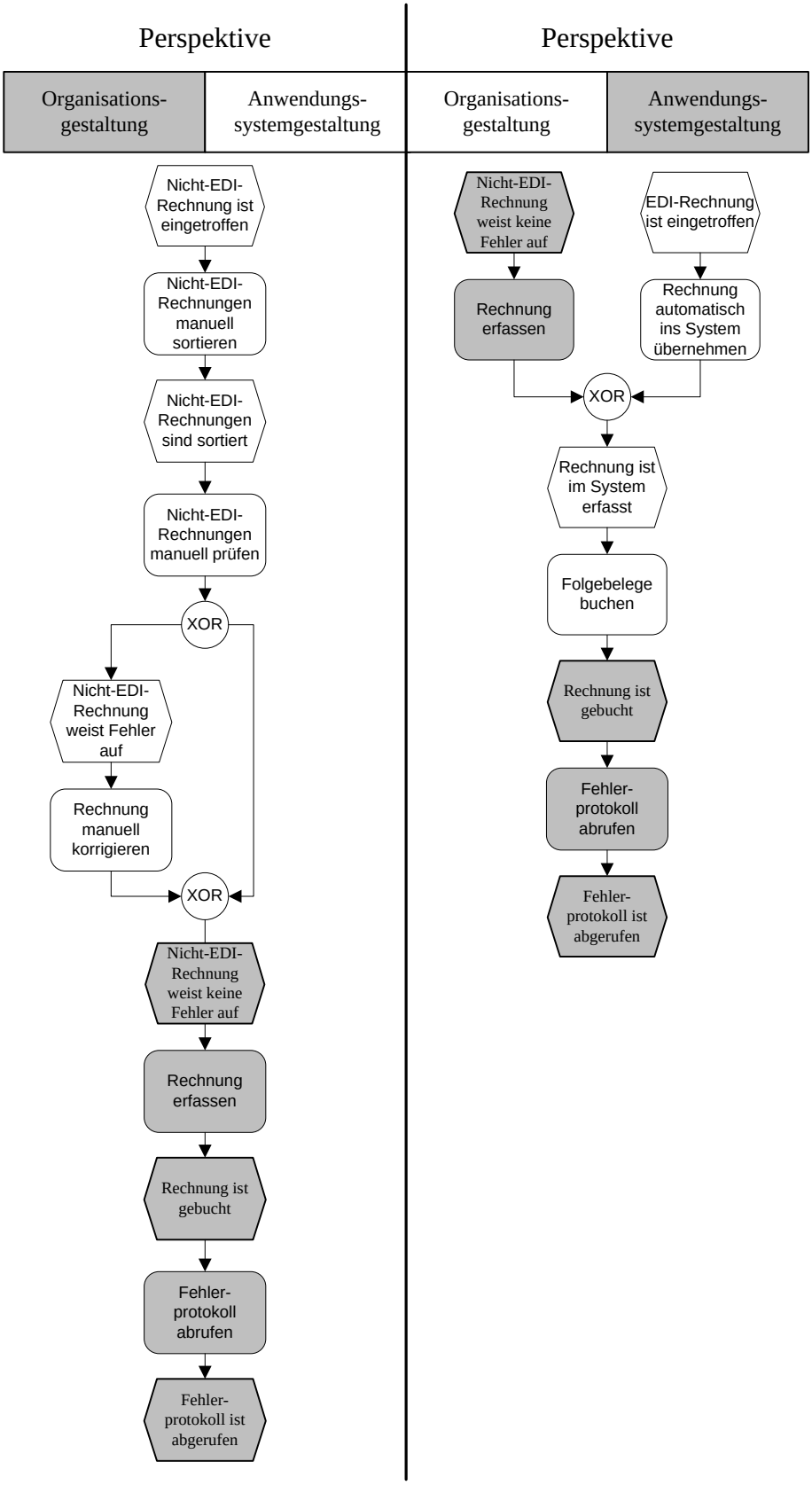



**Abbildung 16: Perspektivenkonfiguration feingranular [BHKS00, 102]**

Zur Durchführung der konfigurativen Referenzmodellierung, die sich mit der Konstruktion von Referenzmodellen befasst, schlagen [BDKK02, 34ff] das in Abbildung 17 dargestellte Phasenmodell vor:

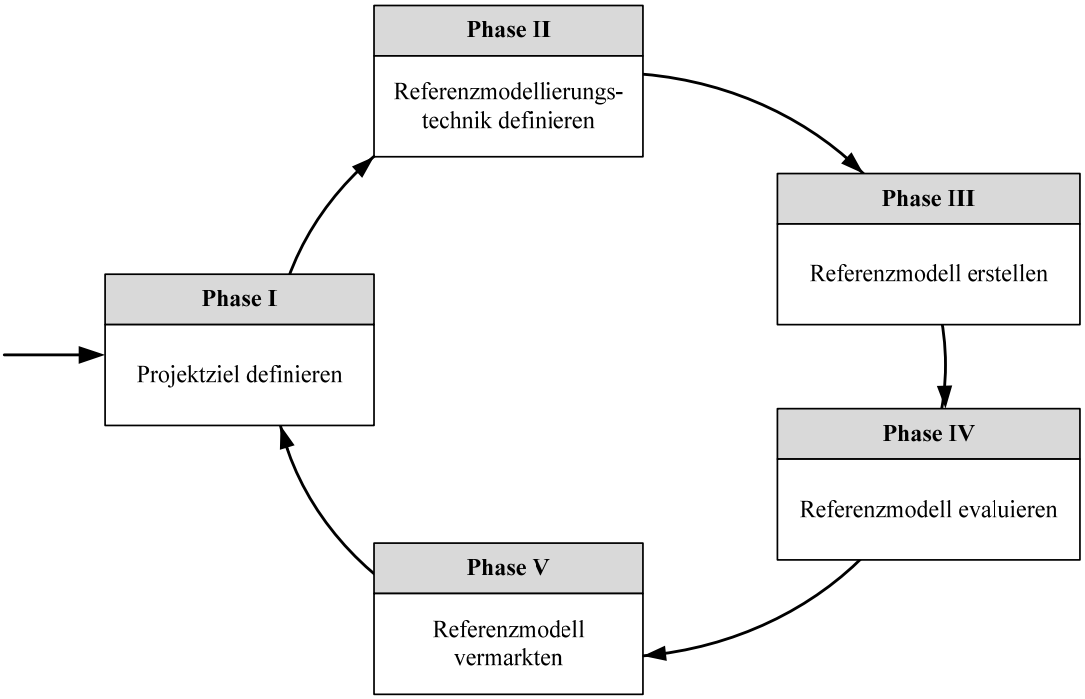



**Abbildung 17: Phasenmodell [BDKK02, 36]**

- *Projektziel definieren:* In dieser Phase wird der für die Referenzmodelle relevante Markt abgegrenzt, die notwendigen Funktionsbereiche, Unternehmensmerkmale und Perspektiven definiert. Durch eine Anforderungsanalyse wird erhoben, welche letztendlich tatsächlich zu berücksichtigen sind. Zusätzlich können die ausgewählten *Determinanten* zu einer Aufwandsschätzung hinzugezogen werden.

- *Referenzmodellierungstechnik definieren:* Eine Referenzmodellierungstechnik kann mehrere übliche Modellierungstechniken kombinieren, z.B. können als Referenzmodellierungstechnik sowohl die Modellierungstechniken Entity-Relationship-Modelle (ERM) und ereignisgesteuerte Prozessketten (EPK) verwendet werden [BDKK02, 43]. Unter Modellierungstechniken werden hier somit Prozessmodellierungsmethoden verstanden, von denen bereits ausgewählte in Abschnitt 2.1.4 vorgestellt wurden.

  Es müssen folgende drei Sichten zur der Definition der Referenzmodellierungstechnik bezüglich der Methode berücksichtigt werden [Holt00, 4ff]:

  - *Konzeptioneller Aspekt:* Durch diesen Aspekt werden die Bedeutung der Begriffe und der Beziehungen der zugrundegelegten Modellierungstechnik, also die semantische Bedeutung der verwendeten Begriffe berücksichtigt, z.B. wird bei EPK zwischen den Begriffen „Ereignis“, „Funktion“ und „Operatoren“ unterschieden und ihre Bedeutung definiert.

  - *Repräsentationeller Aspekt:* Durch diesen Aspekt werden den Begriffen graphische Symbole zugeordnet. Es wird festgelegt, wie die Begriffe graphisch dargestellt werden (vgl. Abschnitt 1.3).

  - *Struktur und Verhalten:* Durch den konzeptionellen und repräsentationellen Aspekt wird die Struktur der Organisation und der Anwendungssysteme dargestellt. Das Verhalten wird durch Handlungsanleitungen dargestellt, das

bedeutet, dass die Anordnungsreihenfolge der *„konkreten Ausprägungen der durch die Orthosprache definierten Begriffe“* festgelegt wird [BDKK02, 45]. Durch die nach den Metamodellen gültigen Anordnungen kann die Struktur der Organisation und der Anwendungssysteme dargestellt werden. Die Reihenfolge der Ausführungen von Tätigkeiten wird durch das Verhalten bestimmt.

Inhaltlich sind für die konfigurative Referenzmodellierung folgende Komponenten zu spezifizieren [BDKK02, 47f]:

- *Modellrahmen:* Der Modellrahmen wird durch die Verwendung von Ordnungsrahmen geschaffen. Diese bieten eine Hilfestellung für das Referenzdesign und gliedern das Modell in die Gestaltungsfelder *Kernprozesse*, *Supportprozesse* und *Führungsprozesse* (Koordinationsprozesse) sowie in Gestaltungsfelder für das Unternehmens- bzw. Unternehmensklassenumfeld (vgl. Abbildung 18). Kernprozesse sind Geschäftsprozesse, die durch einen hohen Anteil an der Wertschöpfung gekennzeichnet sind und einen erheblichen Beitrag zur Unternehmensleistung beisteuern (z.B. Auftragsbearbeitung, Produktion, Distribution); Supportprozesse oder auch Unterstützungsprozesse unterstützen die Leistungserstellung und *„sind in der Regel nicht wettbewerbskritisch“* (z.B. Kostenrechnung); Führungsprozesse oder auch Steuerungsprozesse dienen dazu, die *„Gesamtheit der Geschäftsprozesse“* zu integrieren und ihr Zusammenspiel zu koordinieren (z.B. Strategieentwicklung, Führung) [Gada05, 40]. Durch die Einbeziehung des Umfelds werden somit sowohl die Lieferantenseite als auch die Abnehmerseite in die Referenzmodelle mit einbezogen.

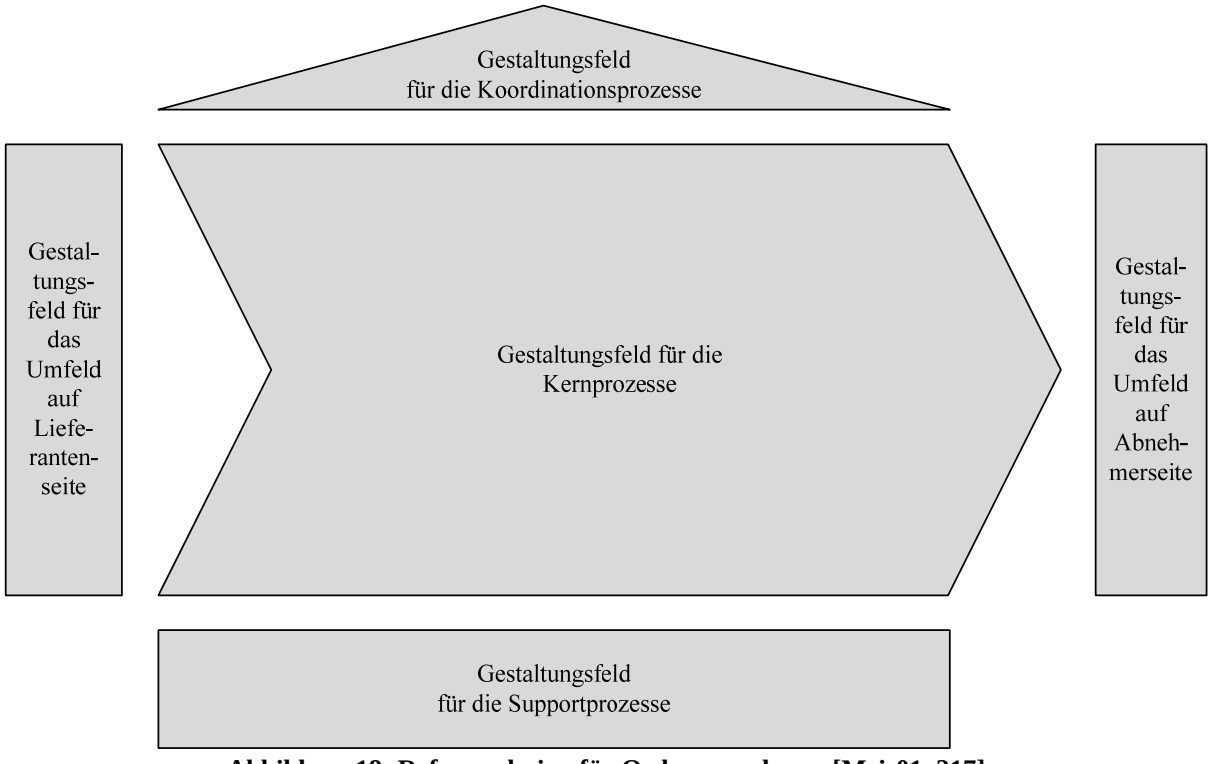



**Abbildung 18: Referenzdesign für Ordnungsrahmen [Meis01, 217]**

Die Prozesse können aufgrund der hohen Abstraktionsebene entweder durch Wertschöpfungskettenelemente [BHKS00, 99f] oder als Prozessauswahlmatrizen dargestellt werden (vgl. zu Prozessauswahlmatrizen [Schü98, 226ff]).

- *Verfeinerungsmodelle:* Den einzelnen Elementen des zuvor erstellten Ordnungsrahmen werden Verfeinerungsmodelle zugeordnet, die unterschiedliche Stufen aufweisen können, um die Übersichtlichkeit zu erhöhen. Laut [BHKS00, 99f] ist es üblich als Verfeinerung ereignisgesteuerte Prozessketten zu verwenden.

- *Konfigurationsregeln:* Durch Konfigurationsregeln wird bestimmt, wie die konfigurierbaren Referenzmodelle *„über explizite Anpassungspunkte“* an die ausgewählten Unternehmensmerkmale und Perspektiven angepasst werden sollen. Dabei müssen die eingangs erwähnten methodischen Komponenten – der konzeptionelle und der repräsentationelle Aspekt – berücksichtigt werden. [BDKK02, 48ff] entwickelten Metamodelle zur *„Konfiguration von Referenzmodellen“* und berücksichtigen dadurch den konzeptionellen Aspekt. Zur Konfiguration des repräsentationellen Aspekts werden von [Schü98, 240ff] Buildtime-Operatoren oder von [Schw99, 144] annotierte Parametrisierungsregeln vorgeschlagen.

- *Referenzmodell erstellen:* In dieser Phase wird der Ordnungsrahmen, die dazugehörigen Verfeinerungsmodelle, die Konfigurationsregeln entwickelt. Es wird dabei auf die in Phase I (Projektziel definieren) definierten Funktionsbereiche sowie Perspektiven und Unternehmensmerkmale zurückgegriffen.

- *Referenzmodell evaluieren:* Bereits während der Erstellung des Referenzmodells soll mit dem Testen begonnen werden. Dies kann durch die Verifikation der Modelle gegen die – gemäß der eingesetzten Modellierungsmethode – verwendeten Metamodelle erfolgen. Zusätzliche inhaltliche Prüfungen sollen durch an den Prozess beteiligten Fachanwender durchgeführt werden. Dabei soll die Richtigkeit folgender Abhängigkeiten der Modelle überprüft werden: Konfigurationsdependenz, Funktionsbereichsdependenz und Domänenwissendependenz. Durch die *Konfigurationsdependenz* soll überprüft werden, ob durch die Konfigurationsregeln geeignete Modellvarianten abgeleitet werden. Durch die *Funktionsbereichsdependenz* soll die inhaltliche und funktionsbereichsübergreifende Konsistenz geprüft werden. Die *Domänenwissensdependenz* soll durch Dritte geprüft werden. Hier wird geprüft, ob die *„betriebswirtschaftlichen Gestaltungsempfehlungen und die Hypothesen bezgl. der Benutzeranforderungen“* zu den Konfigurationsregeln konsistent sind.

- *Referenzmodell vermarkten:* Für die Vermarktung eines Referenzmodells schlägt [BDKK02, 56f] vor, *„sämtliche Dimensionen eines Marketingmix“* nach [Meff91] zu berücksichtigen: Distributionspolitik, Produktpolitik, Kommunikationspolitik und Kontrahierungspolitik. Für eine ausführliche Erklärung dieser Dimensionen sei hier auf erwähnte Literatur verwiesen, da dies nicht Gegenstand dieser Diplomarbeit ist. Die Autoren erwähnen die Notwendigkeit nach einer engen Beziehung zwischen Referenzmodellnutzern und Referenzmodellherstellern, um aus den von den Referenzmodellnutzern gemachten Erfahrungen zu lernen.

Eine wichtige *„Querschnittaufgabe der Referenzmodellierungsmethodik“*, die durchaus eine Herausforderung darstellen kann, ist das *Komplexitätsmanagement*. Es wird versucht, der Komplexität der Referenzmodelle durch Komplexitätsreduktion und Komplexitätsbeherrschung Herr zu werden.

Durch Regeln können konfigurierbare Referenzmodelle an einen projektspezifischen Anwendungskontext angepasst werden, und durch automatisierte Konfigurationsmechanismen kann der Aufwand von Anpassungen trotz einer hohen Wiederverwendungsreichweite gering gehalten werden [BDKK02, 26]. [Schü98, 316] teilt die *„Anwendung konfigurierbarer Referenzmodelle“* in folgende zwei Phasen:

- ***Konfiguration:*** Durch Auswahl der gewünschten Ausprägungen der vorhandenen Konfigurationsparameter eines Referenzmodells wird durch Regelauswertung eine Modellvariante abgeleitet.

- ***Anpassung:*** In dieser Phase werden vom Modellanwender manuell Anpassungen an der erhaltenen Prozesslösung vorgenommen, die sich aus dem Anwendungskontext ergeben. Auch in konfigurierbaren Referenzmodellen können nicht alle Besonderheiten eines Anwendungskontexts modelliert werden.

Durch *Konfigurationsmechanismen* werden Anpassungen koordiniert an expliziten Anpassungspunkten durchgeführt [BeKn02, 42]. Für eine ausführliche Erklärung der Konfigurationsmechanismen sei hier auf [BDKK02, 92ff] verwiesen.

### 2.2.2 Reaktive Wiederverwendung

In Abschnitt 2.2.2.1 werden zuerst von allen ausgewählten Ansätzen gemeinsam verwendete Konzepte zur Wiederverwendung herausgefiltert. Anschließend wird kurz auf die Gemeinsamkeiten zwischen den Ansätzen der musterbasierten Klasse (Abschnitt 2.2.2.2), der bausteinbasierten Klasse (Abschnitt 2.2.2.3) und der fallbasierten Klasse (Abschnitt 2.2.2.4) eingegangen.

#### *2.2.2.1 Gemeinsame Konzepte*

Die reaktive Wiederverwendung hat sich zum Ziel gesetzt, Wissen der Experten explizit zu machen, um immer wiederkehrende *Probleme*, deren *Lösungen* erfolgreich eingesetzt werden, mit den bereits vorgefertigten Prozesslösungen schneller lösen zu können. Dies erfolgt in den ausgewählten Ansätzen unterschiedlich (vgl. Abschnitte 3, 4 und 5).

Abbildung 19 zeigt die Vorgehensweise der reaktiven Wiederverwendung im Überblick. Die Wissensbasis gilt als Grundlage für das anschließende Suchen, in der das explizite Prozesswissen der Experten abgespeichert wird. Dazu müssen die Prozessmodelle für die Wissensbasis aufbereitet und als Artefakte dem jeweiligen Ansatz entsprechend gespeichert werden (1). Diese Artefakte lösen immer ein bestimmtes Problem in einem bestimmten Zusammenhang (Kontext). Der Prozess der reaktiven Wiederverwendung beginnt mit der Suche nach einem Artefakt aus der Wissensbasis, das das neu aufgetretene Problem lösen soll (2). Wird ein lösendes Artefakt gefunden, wird es wiederverwendet und muss eventuell an den neuen Zusammenhang (Kontext) angepasst werden (3). Wird kein lösendes Artefakt gefunden, muss das Prozessmodell neu erstellt werden. Anschließend muss das Prozessmodell

auf Konsistenz überprüft werden (4). Der Prozess der reaktiven Wiederverwendung endet mit der Nutzbarmachung eines oder mehrerer Artefakte des neu erstellten Prozessmodells (5).

Somit weisen alle Ansätze folgende Konzepte auf:

- *Integration von externen Prozesslösungen (Wissenserwerb)*: Externe Prozesslösungen werden für die Wissensbasis entsprechend aufbereitet.
- *Suche nach geeigneten Prozesslösungen*: In der Wissensbasis wird zu einem aktuellen Problem (einer aktuellen Anforderung) nach einer geeigneten Prozesslösung gesucht, die dieses Problem lösen soll.
- *Anwendung von gefundenen Prozesslösungen*: Die gefundenen Prozesslösungen werden angewendet. Dazu zählen auch Strategien der Anpassungen, wenn die gefundenen Prozesslösungen das aktuelle Problem nicht vollständig lösen oder zu einer Gesamtheit zusammengesetzt werden müssen, um das Problem zu lösen.
- *Überprüfung der angepassten Prozesslösungen*: Nach vorgenommen Anpassungen an den Prozesslösungen sollen diese überprüft werden, um Inkonsistenzen zwischen diesen aufdecken zu können.
- *Integration neuer Prozesslösungen in die Wissensbasis*: Die neue Prozesslösung wird für die spätere Wiederverwendung in der Wissensbasis abgelegt.

#### *2.2.2.2 Musterbasierte Ansätze*

Zu der Klasse der musterbasierten Ansätze zählen beispielsweise Hagen [Hage05], Gnatz et al. [GMPR01a], Coplien [Copl96], Ambler [Ambl98], Störrle [Stör01] und Noble [Nobl98].

Jeder dieser Ansätze unterstützt den Prozess der reaktiven Wiederverwendung unterschiedlich. Doch weisen laut Hagen alle Definitionen dieses Prozessmusterbegriffs folgende Elemente auf [Hage05, 10]:

- *Problem:* Jedes Prozessmuster löst ein bestimmtes Problem. Probleme geben Aufschluss darüber, ob das Prozessmuster geeignet ist.
- *Kontext:* Mit dem Kontext werden Bedingungen eines Prozessmusters beschrieben, die vor und nach der Anwendung auftreten müssen, um das Prozessmuster anwenden zu können.
- *Lösung:* Die Lösung löst schließlich das Problem und hat sich bereits bewährt. Mit der Lösung können beispielsweise Prozessschritte angeführt werden, um das Problem zu lösen.

In der Wissensbasis werden bei diesen Ansätzen Prozessmuster abgespeichert, nach denen bei Bedarf gesucht wird und dessen Lösung in weiterer Folge wiederverwendet und angepasst wird.

Die erwähnten Ansätze sind unterschiedlich mächtig. Manche formalisieren die verwendeten Konzepte zur reaktiven Wiederverwendung nicht, womit eine Verwendung dieser nicht eindeutig definiert ist [Hage05]. Beispielsweise werden die Beziehungen zwischen den Prozessmustern nicht formalisiert, was eine Wiederverwendung erschwert, da die Anordnung

der Prozessmuster nicht eindeutig ist (vgl. [GMPR01b], [Ambl98], [Stör00], [Nobl98]) oder Prozessmuster stellen keine Lösungen in Form von schrittweisen Beschreibungen zur Verfügung [Copl94].

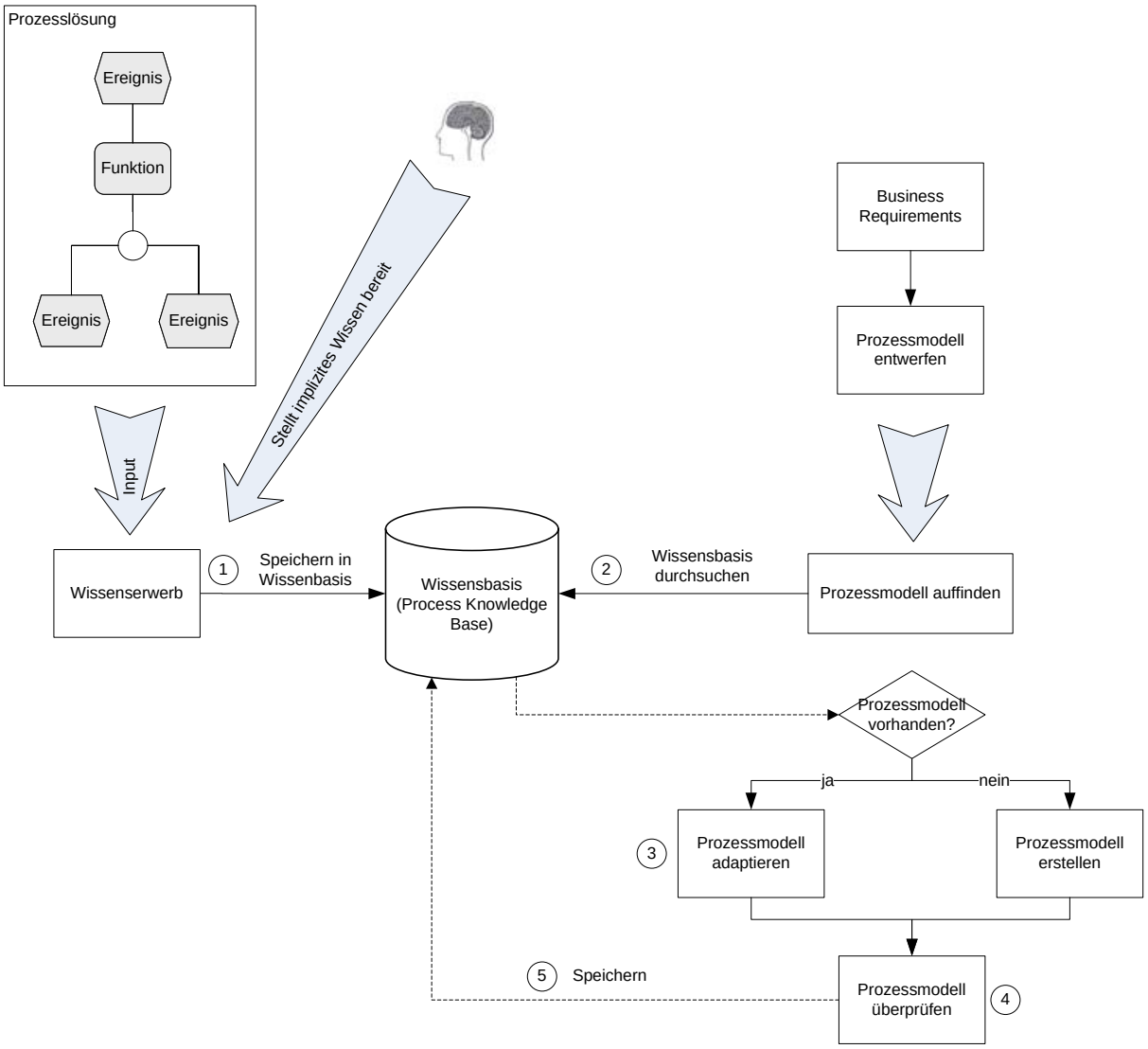



**Abbildung 19: Grundlegender Aufbau der reaktiven Wiederverwendung**

### *2.2.2.3 Bausteinbasierte Ansätze*

Zu dieser Klasse des Modellierungsansatzes zählen die Ansätze nach Rupprecht [Rupp02], Lang [Lang97] und Remme [Remm97].

Die bausteinbasierten Ansätze verwenden Prozessbausteine, die Prozesslösungen zur Verfügung stellen und – analog zu den musterbasierten Ansätzen – ein bestimmtes Problem in einem bestimmten Zusammenhang (Kontext) lösen. Diese Prozessbausteine werden in einer Wissensbasis gespeichert und können bei Bedarf wiederverwendet werden. Dabei gibt es auch bei diesen Ansätzen Unterschiede bei der Umsetzung des Prozesses der reaktiven Wiederverwendung.

#### *2.2.2.4 Fallbasierte Ansätze*

„Fallbasiertes Schließen“, auch Case Based Reasoning (CBR) genannt, ist eine Problemlösungsmethode, die spezifisches Wissen bereits gelöster Problemsituationen nutzt. Mit dieser Methode wird nach ähnlichen Problemen gesucht und die gefundenen Lösungen wieder verwendet und an den neuen Anwendungskontext (Problem) angepasst. Ein besonderes Merkmal dieses Ansatzes ist, dass aus bereits gemachten Erfahrungen kontinuierlich gelernt wird und daher die Modellierung des neuen Problems erleichtert wird. [Kram98, 45]

Dies geschieht in Anlehnung an das menschliche Problemlösungsverfahren mit den Aufgaben Erinnern an die gemachten Erfahrungen, Anpassen der Erfahrungen an die neue Problemsituation, Indizieren dieser Erfahrungen und Reorganisation in der Gedächtnisstruktur zur späteren Wiederverwendung [Rich03, 408].

Bei dieser Klasse von Ansätzen wird ein *Fallbeispiel* oder *Fall* benötigt, der die Problemsituation (Anwendungskontext) beschreibt und die bei der Lösung des Falles gesammelten Erfahrungen beinhaltet. Diese Fälle werden in einer *Fallbasis*, auch *Fallbibliothek* gespeichert und sollen für eine Suche geeignet strukturiert werden. Durch *Fallbasiertes Schließen* wird somit ein neu auftretendes Problem durch Erinnerung an eine bereits gelöste Problemsituation und Anpassung der Erfahrung an den neuen Anwendungskontext gelöst. [Wess96]

Daraus leiten Althoff und Bartsch-Spörl folgende Merkmale für das fallbasierte Schließen ab [AlBa96]:

- Gefundene Fälle müssen an den neuen Kontext des Problems angepasst werden. Es werden effektive Anpassungsstrategien benötigt.
- Es müssen oft mehrere Fälle kombiniert werden, da die Probleme häufig komplex sind.
- Allgemeinwissen spielt bei der Auswahl von geeigneten Fällen und Anpassung eine sehr große Rolle.
- Bei ständig neu auftretenden Situationen, für das noch kein Modell vorhanden ist, kann das fallbasierte System nur Prozesslösungen anbieten, die jedoch vom Anwender überprüft werden müssen.

Ein weit verbreitetes Ablaufmodell für das fallbasierte Schließen liefern Aamodt und Plaza [AaPl94, 6f], die den CBR-Cycle (Abbildung 20) mit den folgenden vier Phasen entwickelt haben:

***Retrieve:*** In dieser Phase werden aus einer geeignet organisierten Fallbibliothek passende Fälle für ein neu aufgetretenes Problem gesucht. Dies geschieht über die Bestimmung des Ähnlichkeitsmaßes in der Problembeschreibung zwischen der der vorhandenen Fälle und der des neuen Problems (*New Case*). Die spätere Anpassung des/der *Retrieved Case(s)* an die aktuelle Problemsituation erfolgt durch eine geeignete Strategie.

***Reuse:*** In dieser Phase wird die bereits vorhandene, in dem fallbasierten System dokumentierte Erfahrung zur Lösung des aktuellen Problems wieder verwendet, nachdem die Lösung an den Kontext des aktuellen Problems angenähert wurde (*Suggested Solution*).

***Revise:*** Diese Phase hat zur Aufgabe, die Suggested Solution in eine reale Umwelt zu überprüfen. Das Ergebnis der Untersuchung wird an das fallbasierte System übergeben und die Suggested Solution wird zu einer *Confirmed Solution*, womit eine Aussage über die Bewährung der Prozesslösung getroffen wird.

***Retain:*** Der neue Fall wird nun in dieser Phase als *Learned Case* in der Fallbibliothek gespeichert. Sollen die Auswahl- und Anpassungsstrategie optimiert werden, wird dies ebenfalls vermerkt.

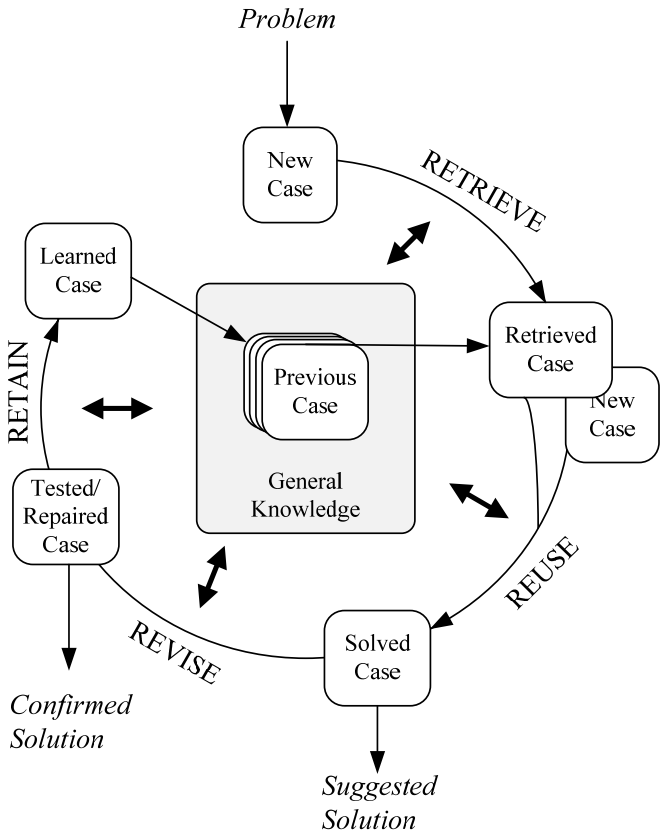



**Abbildung 20: CBR-Cycle nach [AaPl94, 7]**

Sowohl Krampe [Kram98] als auch Schulze [Schu01] verwenden den CBR-Cycle nach Aamodt und Plaza [AaPl94] mit den Phasen *Retrieve*, *Reuse*, *Revise* und *Retain*, die jedoch unterschiedlich realisiert werden.

### *2.2.2.5 Zusammenfassung*

In diesem Abschnitt wurde eine kurze Einführung in die gemeinsamen Konzepte der Ansatzklassen der reaktiven Wiederverwendung gegeben. Dabei ist zu erkennen, dass alle Ansätze das Wissen zu Beginn erwerben und in einer Wissensbasis abspeichern müssen. Je nach Ansatzklasse wird das Prozesswissen als Artefakte gespeichert, und zwar entweder als Prozessmuster, Prozessbausteine oder Fallbeispiele. Die reaktive Wiederverwendung beginnt mit der Suche nach gespeicherten Artefakten. Werden geeignete Artefakte gefunden, werden sie an die neue Problemsituation angepasst und anschließend auf Korrektheit überprüft. Die reaktive Wiederverwendung endet mit dem Abspeichern der angepassten Lösung in der

Wissensbasis, um die angepasste Lösung für eine spätere Wiederverwendung nutzbar zu machen.

Die ausgewählten Ansätze werden anhand der Konzepte *Integration von externen Prozesslösungen, Suche nach geeigneten Prozesslösungen, Anwendung von gefundenen Prozesslösungen, Überprüfung der angepassten Prozesslösungen* und *Integration neuer Prozesslösungen in die Wissensbasis* in den Abschnitten 3, 4 und 5 untersucht.

# 3 Musterbasierter Ansatz nach Hagen

Hagen beschreibt in ihrer Arbeit die Prozessmusterbeschreibungssprache PROPEL (Process Pattern Description Language). PROPEL ist als eine Erweiterung von UML (Unified Modeling Language) zu verstehen und baut auf den Modellierungskonzepten der UML-Aktivitätsdiagramme auf. UML ist analog zu EPK eine semi-formale Modellierungssprache. Hagen erweitert die UML-Aktivitätsdiagramme um eine formale Syntax und formale Semantik, um das Problem der fehlenden Formalisierung zu beheben. [Hage05, 6, 69ff]

Dieser Ansatz verwendet, wie der Name schon sagt, im Rahmen der reaktiven Wiederverwendung Prozessmuster, um *„einen bewährten Prozess“* zu beschreiben, der *„ein Problem, das in einem bestimmten Kontext wiederholt aufgetreten ist“*, lösen soll [Hage05, 73].

Durch eine einheitliche Beschreibung werden sowohl *Probleme* als auch *Prozessmuster* einheitlich dokumentiert werden (vgl. Problem- und Prozessmusterschemata Abschnitt 3.4.1) und können durch Diagramme (Problem- und Prozessmusterdiagramme) abgebildet werden [Hage05, 70]. Ferner werden Probleme und die lösenden Prozessmuster mit Hilfe der *Beziehungen* strukturiert in einem *Prozessmusterkatalog*, der als Wissensbasis dient, zu einem bestimmten Themengebiet zusammengefasst [Hage05, 47], wobei jedem Prozessmuster genau ein Prozessmusterkatalog zugeordnet wird [Hage05, 74]. Somit können mehrere Prozessmusterkataloge als Wissensbasis existieren [Hage05, 94]. Die Zuordnung zwischen Problem und dem lösenden Prozessmuster erfolgt über den initialen und den resultierenden *Kontext* [Hage05, 50f]. Ein Problem kann von keinem oder mehreren Prozessmustern gelöst werden, wobei diese Prozessvarianten darstellen, da sie dasselbe Problem lösen [Hage05, 76]. Die Vorgehensweise zur Dokumentation wird in Abschnitt 3.4.1 erläutert.

Die Suche nach in Frage kommenden Prozessmustern erfolgt über die Identifizierung eines Problems, indem der initiale und der resultierende Kontext definiert werden. Über die Zuordnung zwischen Problem und Prozessmuster werden die lösenden Prozessmuster identifiziert. Durch die Anordnung der Prozessmuster in einem Prozessmusterkatalog können über die Prozessmusterbeziehungen weitere Prozessmuster identifiziert werden. Dabei kann entweder die Top-Down- oder From-Within-Strategie verwendet werden (vgl. Abschnitt 3.4.2).

Zur Anpassung der Prozessmuster an den aktuellen Kontext wird keine Unterstützung durch diesen Ansatz geboten. Der Pattern User kann zusätzlich benötigte Details manuell hinzufügen (vgl. Abschnitt 3.4.3).

Soll die neue Lösung ebenfalls in die Wissensbasis aufgenommen werden, erfolgt die Dokumentation der Prozessmuster und ihrer Beziehungen zu den bereits bestehenden Prozessmustern und untereinander manuell (vgl. Abschnitt 3.4.5).

Die formale Syntax der eingeführten Konzepte wird durch OCL-Regeln (Object Constraint Language) – analog zur UML-Spezifikation [UML1.5], durch sogenannte Constraints eingeschränkt [Hage05, 72]. Desweiteren wird durch die formale Syntax und Semantik verhindert, dass Prozessmuster öfter abgespeichert werden und redundant in der Wissensbasis enthalten sind.

Bevor die Vorgehensweise der Wiederverwendung detailliert anhand des Rahmenbeispiels erklärt wird, werden zuerst die von diesem Ansatz verwendeten Konzepte erläutert.

## *3.1 Prozessmuster*

Ein Prozessmuster setzt sich somit aus den Grundelementen Problem, Kontext und Lösung zusammen (Abbildung 21).

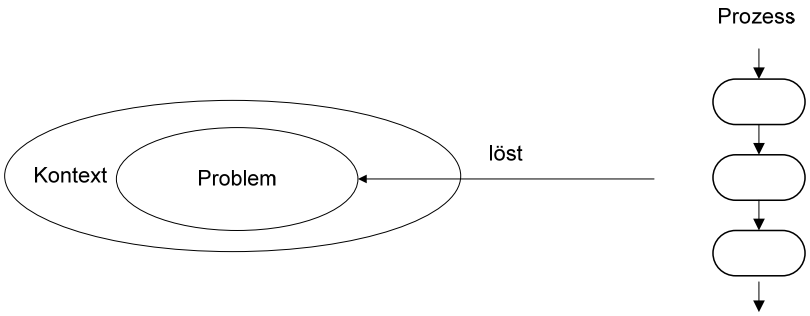



**Abbildung 21: Elemente eines Prozessmusters [Hage05, 48]**

### Problem

Ein *Problem*, das durch das Prozessmuster gelöst wird, kann entweder durch einen Fragesatz, einen Aussagesatz oder eine kurze Problembeschreibung beschrieben werden [Hage05, 50]. Beispielsweise kann ein Prozessmuster das Problem „Wie wird der Antrag bearbeitet?“ lösen.

### Kontext

Um es jedoch weiter zu präzisieren, werden zu einem Problem zusätzlich der *initiale* und der *resultierende Kontext* angegeben, der jeweils aus einer *„Menge von Objekten und Ereignissen“* besteht. Der Kontext löst das Prozessmuster aus bzw. ist nach der Anwendung des Prozessmusters das Ergebnis des Prozesses. Ein Objekt stellt ein bestimmtes Artefakt dar, das während der Durchführung des Prozesses *„konsumiert, erstellt oder modifiziert“* werden kann. [Hage05, 49f] Ein Beispiel ist das Objekt „Antrag“, das durch das Problem „Wie wird der Antrag bearbeitet?“ konsumiert wird. Ein Ereignis bezeichnet *„ein Geschehen, das einen Prozess auslöst“* oder durch den Prozess erzeugt wird [Hage05, 49]. Beispielsweise ist der initiale Kontext das Ereignis „Antrag erfasst“ und der resultierende Kontext bei dem vorher genannten Problem „Antrag bearbeitet“.

Abbildung 22 zeigt beispielhaft die Kontextdarstellung eines Problems. Das Problem „Wie wird der Antrag bearbeitet?“ hat das Objekt „Antrag“ und das Ereignis „Antrag erfasst“ als initialer Kontext und das Ereignis „Antrag bearbeitet“ als resultierender Kontext. Dabei wird der Text von Objekten einfach und von Ereignissen doppelt unterstrichen und das Problem in einer Ellipse dargestellt [Hage05, 134f].

Zur Definition des Kontextes können die Objekte und Ereignisse auch zusammengesetzt werden („*Komposition von Objekten und Ereignissen*“). Dabei können stets nur Objekte zu einem Objekt und Ereignisse zu einem Ereignis komponiert werden. Durch das Zusammenfassen können Prozesse übersichtlicher dargestellt werden. [Hage05, 49]

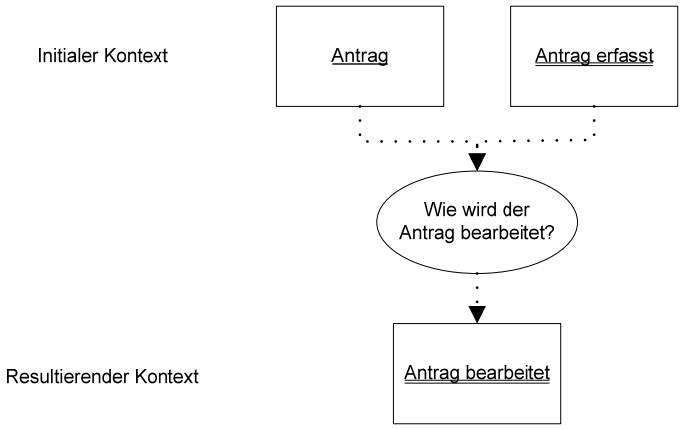



**Abbildung 22: Beispiel Kontext eines Problems**

## Lösung

Die *Lösung* des Prozessmusters sind schließlich die Prozessschritte selbst, die zwischen dem initialen und dem resultierenden Kontext liegen [Hage05, 50]. Das Prozessmuster stellt somit die Lösung für ein bestimmtes Problem in einem bestimmten Anwendungskontext zur Verfügung. Die zu konsumierenden Objekte bzw. die auslösenden Ereignisse lösen als initialer Kontext das Prozessmuster aus und nach der Anwendung dieses Prozessmusters sind die erzeugten Objekte bzw. die erzeugten Ereignisse (resultierender Kontext) das Ergebnis des Prozessmusters.

Abbildung 23 zeigt ein Notationsbeispiel eines Prozessmusters mit zwei initialen Kontexten und einem resultierenden Kontext. Die Bedingungen (Kontext), die vor und nach Anwendung des Prozessmusters erfüllt sein müssen, decken sich mit den Bedingungen (Kontext), die vor und nach dem Problem bestehen (Kontext) [Hage05, 51f]. Das Prozessmuster „Antrag bearbeiten“ löst nun das Problem „Wie wird der Antrag bearbeitet?“, da die Kontexte übereinstimmen.

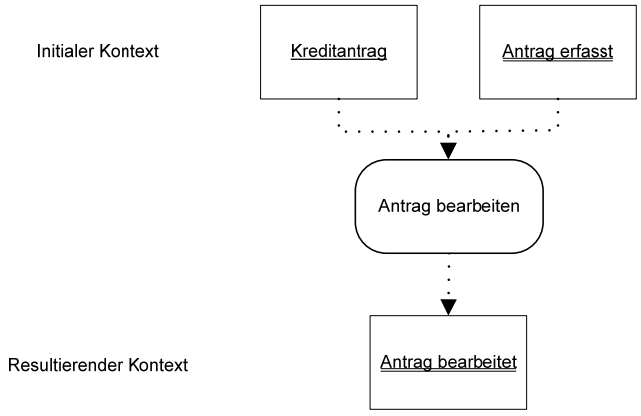



**Abbildung 23: Beispiel Kontext eines Prozessmusters**

Eine Suche nach geeigneten Prozessmustern wird durch eine Einteilung in die drei Klassifikationsmerkmale Domäne, Phase und Aspekt erleichtert (Abbildung 24). Durch die *Domäne* wird bestimmt, in welchen fachlichen Einsatzbereich das Prozessmuster einzuordnen ist, wie z.B. Softwareentwicklung oder Testen. Da sich Hagen in ihrem Ansatz nur mit der Domäne Softwareentwicklung beschäftigt, bezieht sich das Merkmal Phase rein auf Phasen der Softwareentwicklung. In der *Phase* wird festgehalten, in welchem Abschnitt des Projekts bzw. der Tätigkeit das Prozessmuster zum Einsatz kommt. Als Beispiel werden die Phasen des RUP-Konzepts erwähnt: Konzeption, Entwurf, Realisierung und Nutzungsübertragung.

Das Merkmal *Aspekt* beschreibt die thematische Ausrichtung, wie z.B. Design, Qualitätssicherung oder Risikomanagement. [Hage05, 46]

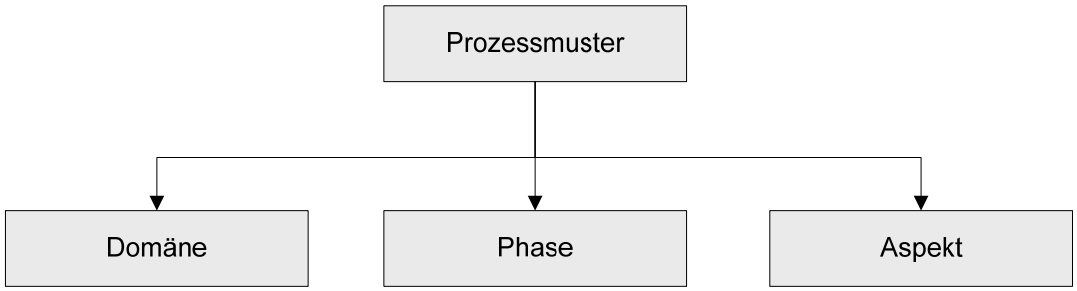



**Abbildung 24: Klassifikationsmerkmale von Prozessmustern**

Durch die Einführung von *Prozessmusterschema* und *Problemschema* werden durch definierte Elemente Prozessmuster und Probleme einheitlich beschrieben [Hage05, 65ff].

Das von Hagen definierte Prozessmusterschema enthält die in Tabelle 2 dargestellten Elemente. Die Begriffe *Domain*, *Phase*, *Aspect* und *Catalog* (Prozessmusterkatalog) wurden bereits erklärt. *Version* bezieht sich auf die Versionierung des Prozessmusters. Dies wurde bei diesem Ansatz eingeführt, um bei Veränderungen nicht neue Prozessmuster mit neuen Namen generieren zu müssen, sondern eine Verknüpfung zwischen abgeleiteten Prozessmustern herstellen zu können. *Synonyms* enthalten alle Namen für dieses Prozessmuster. *Problem* enthält einen Verweis auf das Problemschema. Die Aktivitäten, die in dem Prozess durchgeführt werden, um zur Lösung zu gelangen, werden in dem Element *Process* beschrieben. *Role* definiert die verantwortliche Rolle für die Ausführung des Prozesses. Sie wird Prozessen und Aktivitäten zugeordnet und kann beliebig viele, aber auch keine Prozesse und Aktivitäten haben. *Related Patterns* beinhaltet die Prozessmuster, die in Beziehung zu diesem Prozessmuster stehen (vgl. Prozessmusterbeziehungen). *Example* enthält ein Beispiel für das Prozessmuster (Instanziierung). Das Attribut *Discussion* beinhaltet die Vor- und Nachteile für die Verwendung des Prozessmusters. [Hage05, 65f]

| Element | Bedeutung |
|---|---|
| Name | Name des Prozessmusters |
| Version | Aktuelle Version |
| Domain | Einordnung in den Einsatzbereich |
| Phase | Einordnung in die Phase, in der das Prozessmuster eingesetzt werden kann. |
| Aspect | Zuweisung der thematischen Ausrichtung, z.B. Projektmanagement |
| Catalog | Einordnung in den Prozesskatalog für das jeweilige Themengebiet |
| Synonyms | Synonyme |
| Problem | Verweis auf ein Problemschema. |
| Process | Notwendige Aktivitäten zur Lösung des Problems. |
| Role | Verantwortlicher für die Ausführung. |
| Related Patterns | Angabe von Prozessmustern, die in Beziehung mit diesem Prozessmuster stehen. |
| Example | Beispiel für Prozessmuster (Instanziierung). |
| Discussion | Beschreibung der *„Motivation, Konsequenzen und Pro- und Kontraargumente“*, die mit dem Prozessmuster verknüpft sind. |

**Tabelle 2: Prozessmusterschema [Hage05, 66]**

Im Prozessmusterschema ist ein Verweis auf das Problemschema enthalten. Das Problemschema ist in Tabelle 3 dargestellt. Es enthält einen *Namen*, eine aktuelle *Version*, den *initialen* und den *resultierenden Kontext*, eine natürlichsprachige Beschreibung und ebenfalls einen Verweis auf Prozessmuster, die dieses Problem lösen.

| Element | Bedeutung |
|---|---|
| Name | Problemname |
| Version | Aktuelle Version |
| Initial Context | Initialer Kontext; dargestellt mit Hilfe von Ereignissen und Objekten, die als *„Voraussetzung für die Anwendung des Prozessmusters“* zu sehen sind. |
| Resulting Context | Resultierender Kontext; dargestellt mit Hilfe von Ereignissen und Objekten, *„die Ergebnis der Anwendung sind“*. |
| Description | Natürlichsprachige Beschreibung des Problems. |
| Solving Process Patterns | Verweis auf die lösenden Prozessmuster. |

**Tabelle 3: Problemschema [Hage05, 67]**

Der Grund für die Trennung zwischen Prozessmusterschema und Problemschema begründet [Hage05, 50] darin, dass es durchaus möglich ist, ein Problem bereits definiert zu haben, jedoch ein lösendes Prozessmuster fehlt. Andererseits kann es auch zu dem Fall kommen, dass es zu einem Problem mehrere lösende Prozessmuster gibt, die Prozessvarianten darstellen (vgl. Prozessmusterbeziehungen), was durch die Einführung des Elements *Solving Process Patterns* im Problemschema gelöst wird.

Somit gibt es zwischen dem Prozessmuster und dem Problem eine Beziehung. Ein Prozessmusterschema soll genau ein Prozessmuster für ein Problem beschreiben, jedoch kann ein Problem von mehreren Prozessmustern gelöst werden und somit auch einen Verweis auf mehrere Prozessmuster haben.

Der von Hagen entwickelte Ansatz verwendet UML-Aktivitätsdiagramme zur Beschreibung von Prozesslösungen und erweitert das Metamodell um eine formale Syntax und formale Semantik [Hage05, 69ff]. Es wird im Rahmen dieser Untersuchung die Modellierungsmethode eEPK zur Beschreibung von Prozesslösungen verwendet. In folgender Tabelle werden die Modellierungskonstrukte des UML-Aktivitätsdiagramms der eEPK gegenübergestellt:

| **UML-Aktivitätsdiagramm** | **eEPK** |
|---|---|
| Objekt | Informationsobjekt |
| Ereignis | Ereignis |
| Prozess | Prozess |
| Aktivität | Funktion |
| Rolle | Organisationseinheit |

**Tabelle 4: Gegenüberstellung der Modellierungskontrukte (UML-Aktivitätsdiagramme, eEPK)**

Somit können alle für die Untersuchung notwendigen Elemente der Prozesslösungen des UML-Aktivitätsdiagramms auch mit eEPK dargestellt werden.

## *3.2 Prozessmusterbeziehungen*

Hagen definiert in PROPEL Beziehungen zwischen Prozessmustern, und zwar Sequence, Use, Refinement und Processvariance [Hage05, 54ff]. Die einzelnen Beziehungen und ihre Notationen werden kurz erklärt.

**Sequence-Beziehung**

Die Sequence-Beziehung verknüpft Prozessmuster sequentiell, wobei *„ein oder mehrere Prozessmuster“* (Vorgängermuster) vor der *„Anwendung eines nachfolgenden Prozessmusters“* (Nachfolgermuster) angewendet werden. Wird der Kontext eines Prozessmusters betrachtet, so bedeutet dies, dass der initiale Kontext des Nachfolgermusters die Vereinigungsmenge aus allen resultierenden Kontexten der Vorgängermuster besteht. [Hage05, 55]

Für die Sequence-Beziehung gibt es unterschiedliche Notationsmöglichkeiten, wie in Abbildung 25 ersichtlich ist. A stellt eine einfache Sequence-Notation dar, bei der zwei Prozessmuster verknüpft sind. Werden mehrere Vorgängermuster mit einem Nachfolgemuster verknüpft, können als alternative Darstellung B (vereinigte Pfeile) oder C (Kasten) verwendet werden. Gibt es mehrere Nachfolgermuster, kann die Darstellung unter D verwendet werden. Alternative Vorgängermuster sind unter E zu finden. Wird nur eine Teilmenge der Vorgängermuster dargestellt, wird die Notation unter F verwendet. [Hage05, 141]

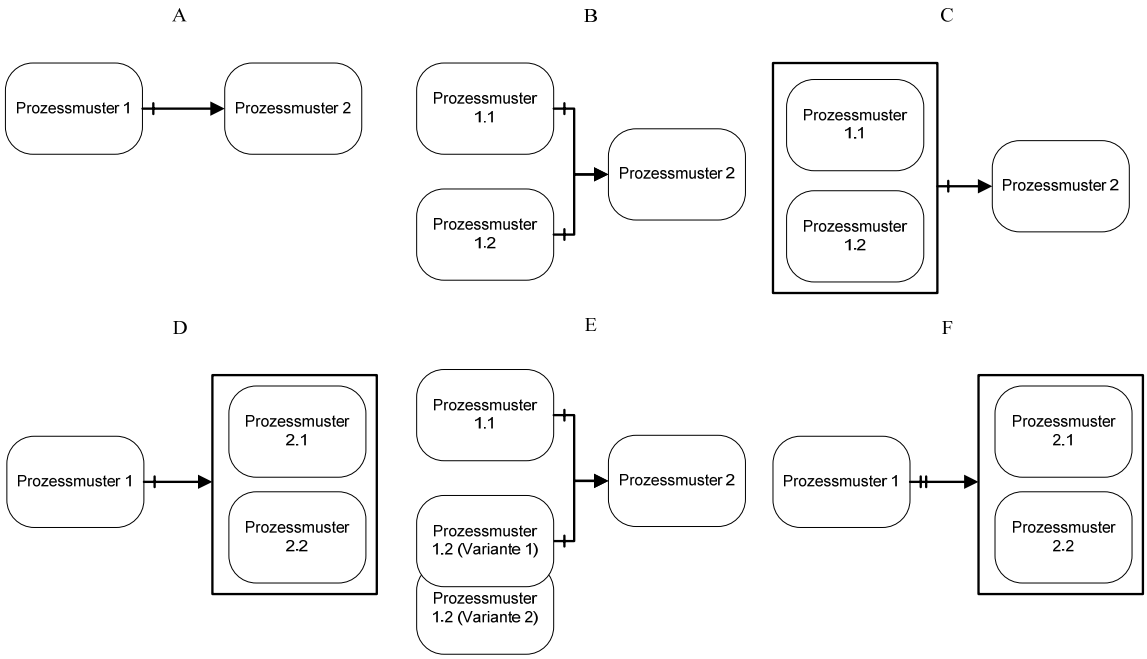



**Abbildung 25: Notationsmöglichkeiten einer Sequence-Beziehung in Anlehnung an [Hage05, 141]**

Die Prozessmuster stehen somit bei einer Sequence-Beziehung stets in einer 1:1-Beziehung. Es wurde keine 1:n-Beziehung, und somit keine Beziehung für Parallelität eingeführt. Hagen begründet dies folgendermaßen: Man betrachte *„ein Vorgängermuster pp, welches drei Objekte $o_1$, $o_2$ und $o_3$ erzeugt. $o_1$ ist Inputobjekt vom Nachfolgermuster pp'$_1$, die Objekte $o_2$ und $o_3$ Inputobjekte vom Nachfolgermuster pp'$_2$. pp steht also jeweils mit pp'$_1$ und pp'$_2$ in einer Sequence-Beziehung“*. [Hage05, 55] Da für diese Situation zwei Sequence-Beziehungen zwischen dem Vorgängermuster und den zwei Nachfolgermuster bestehen, kann die

Notationsmöglichkeit D verwendet werden, das die Nachfolger in einem Kasten zusammenfügt.

**Use-Beziehung**

Durch eine Use-Beziehung dient ein Prozessmuster als Kompositmuster, das beliebig viele weitere Prozessmuster als Komponentenmuster verwenden kann. Die Komponentenmuster stellen Teile des Kompositmusters dar und lösen Teilprobleme des Kompositmusters. Die Gesamtheit aller Komponentenmuster löst das gesamte Problem. Dabei muss eine Übereinstimmung des initialen Kontexts des Kompositmusters mit dem initialen Kontext des ersten Komponentenmusters und des resultierenden Kontexts des Kompositmusters mit dem resultierenden Kontext des letzten Komponentenmusters erfolgen. Das impliziert, dass *„alle Objekte und Ereignisse des Kontexts“* des Kompositmusters in den Komponentenmustern vorkommen müssen. [Hage05, 56] Es wird somit eine Hierarchie geschaffen.

Abbildung 26 zeigt die Notation einer Use-Beziehung unter Verwendung des Kompositmusters und der Komponentenmuster. Es werden zwei Prozessmustersymbole verknüpft, wobei mit dem Prozessmuster begonnen wird, das den gesamten Prozess als Lösung liefert, und die Prozessmuster, die die Teilprobleme lösen, am Ende des befüllten Pfeils dargestellt werden [Hage05, 141]. Das Kompositmuster „Antrag bearbeiten“ benutzt die Prozessmuster „Risiko ermitteln“, „Antrag ablehnen“ und „Antrag bewilligen“. Dabei kann analog zur Sequence-Beziehung ebenfalls ein Kasten zur Zusammenfassung mehrerer Komponentenmuster verwendet werden (vgl. [Hage05, 156]).

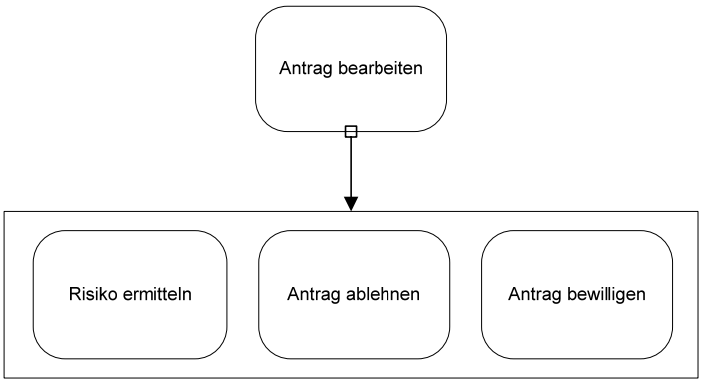



**Abbildung 26: Beispiel Use-Beziehung**

**Refinement-Beziehung**

Durch die Zuordnung zwischen Problem und Prozessmuster können sowohl Prozessmuster als auch Probleme verfeinert werden und durch eine Refinement-Beziehung verknüpft werden [Hage05, 60ff].

Die Refinement-Beziehung zwischen Prozessmuster verknüpft ein abstraktes Prozessmuster (Supermuster) mit dem spezielleren Prozessmuster (Submuster). Das Submuster ist somit eine Spezialisierung des allgemeinen Supermusters. Das Submuster muss mindestens den gleichen initialen und den gleichen resultierenden Kontext aufweisen wie das Supermuster. [Hage05, 59]

Abbildung 27 zeigt ein Beispiel für eine Notation einer Refinement-Beziehung zwischen Prozessmuster. Diese Kante besitzt eine hohle Pfeilspitze am Ende des abstrakten Problems

[Hage05, 142]. Dabei ist das Prozessmuster „Antrag bearbeiten“ das Supermuster und das Prozessmuster „Kreditantrag bearbeiten“ ist die Spezialisierung des Supermusters (Submuster).

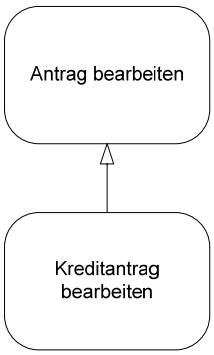



**Abbildung 27: Beispiel Refinement-Beziehung zwischen Prozessmuster**

Wird die Refinement-Beziehung bei Problemen verwendet, so wird hier – analog zur Refinement-Beziehung zwischen Prozessmustern – ein abstraktes Problem (Superproblem) mit einem detaillierten Problem (Subproblem) verknüpft. Das Subproblem ist somit eine Spezialisierung des abstrakten Problems, wobei sowohl alle Objekte als auch Ereignisse im initialen und im resultierenden Kontext des Superproblems auch im initialen und im resultierenden Kontext des Subproblems vorkommen müssen. Falls zur Verfeinerung zusätzliche *„Objekte und Ereignisse“* notwendig sind, können diese im Subproblem ebenfalls angeführt werden. [Hage05, 60]

Abbildung 28 zeigt ein Beispiel einer Refinement-Beziehung zwischen Problemen. Diese Kante besitzt eine hohle Pfeilspitze am Ende des abstrakten Problems [Hage05, 142]. Das Superproblem „Wie wird der Antrag bearbeitet?“ wird von dem Subproblem „Wie wird der Kreditantrag bearbeitet?“ spezialisiert.

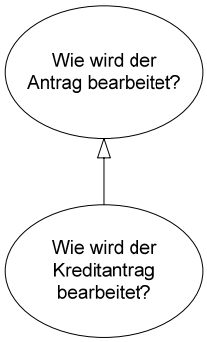



**Abbildung 28: Beispiel Refinement-Beziehung zwischen Problemen**

Das abstrakte Prozessmuster, das eine Refinement-Beziehung zu einem Submuster aufweist, löst ein abstraktes Problem und das spezifische Submuster löst das Subproblem des abstrakten Problems [Hage05, 60f].

### Processvariance-Beziehung

Durch die Processvariance-Beziehung werden Prozessmuster verknüpft, die als Prozessvarianten gelten. Sie lösen *„das gleiche Problem“* mit unterschiedlichen Lösungen. [Hage05, 61]

Abbildung 29 zeigt ein Beispiel für eine Prozessvariance-Beziehung. Diese wird durch eine ungerichtete Kante notiert [Hage05, 143]. Die Prozessmuster „Kreditantrag bearbeiten“ und „Kredit abwickeln“ beinhalten unterschiedliche Prozessschritte zur Lösung des gleichen Problems.

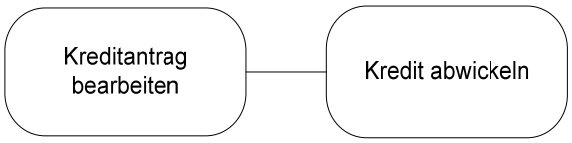



**Abbildung 29: Beispiel Processvariance-Beziehung**

Objekte und Ereignisse können wie bereits erwähnt ebenfalls zusammengesetzt werden (Komposition). Diese Komposition kann die Prozessmusterbeziehungen beeinflussen, da *„Prozessmuster, die dem ersten Anschein nach nicht die Bedingungen zum Zustandekommen einer Beziehung erfüllen“,* durch diese Bedingungen *„die kompositorischen Eigenschaften der Objekte und Ereignisse ihrer Kontexte erfüllen“* können. [Hage05, 62]

## *3.3 Organisation der Wissensbasis*

Die Prozessmuster werden mit ihren Prozessmusterbeziehungen in einem Prozessmusterkatalog abgebildet, der *„das explizierte prozessuale Wissen über ein bestimmtes Themengebiet“* in einem gerichteten Graphen festhält, wobei die Knoten die Prozessmuster und die Kanten die Beziehungen zwischen diesen Prozessmustern darstellen. Der Prozessmusterkatalog stellt somit die Wissensbasis dieses Ansatzes dar. [Hage05, 47ff] Es können mehrere Prozessmusterkataloge vorhanden sein, die den Prozessmustern zugeordnet werden können [Hage05, 66].

Hagen entwickelte in ihrer Arbeit den Prozessmusterkatalog CADS (Catalog for the Development of Software) und verwendete dazu ausgewählte von RUP definierte Prozesse. Diese sind bereits hierarchisch strukturiert, was eine Identifizierung der Use-Beziehungen erleichterte. Aus Gründen der Übersichtlichkeit wurden mehrere Sichten eingeführt, und zwar für jeden Beziehungstyp eine eigene Sicht. Dadurch entstanden die Sichten „Use-Beziehung“, „Sequence-Beziehung“, „Refinement-Beziehung“ und „Processvariance-Beziehung“. [Hage05, 148ff]

Folgende Abbildungen zeigen wie die Sichten eines Prozessmusterkatalogs auf die Sichten der „Use-Beziehung“, „Sequence-Beziehung“, „Refinement-Beziehung“ und „Processvariance-Beziehung“ aufgeteilt werden können.

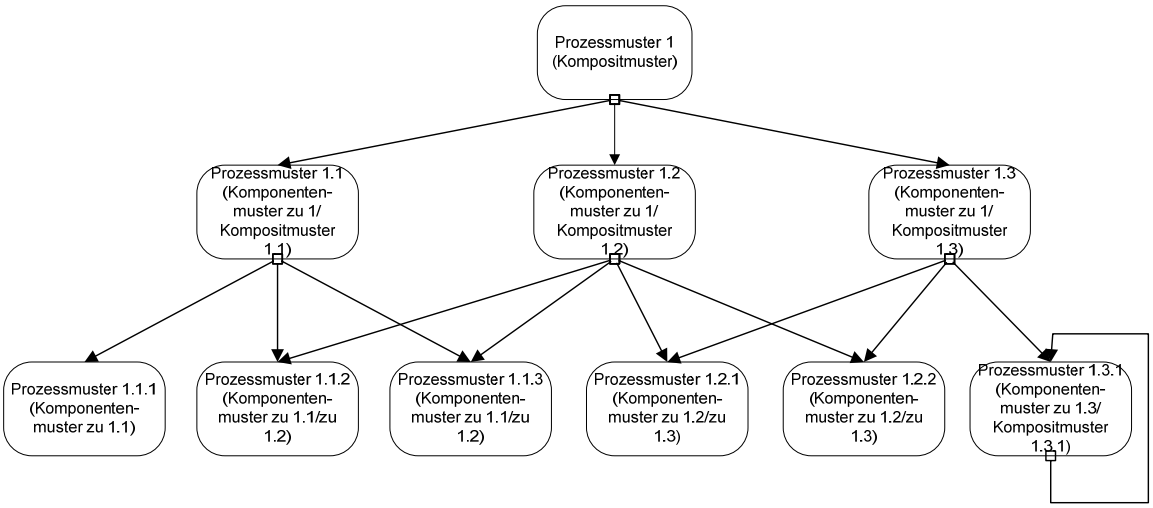



**Abbildung 30: Beispiel Sicht "Use-Beziehung" in Anlehnung an [Hage05, 149]**

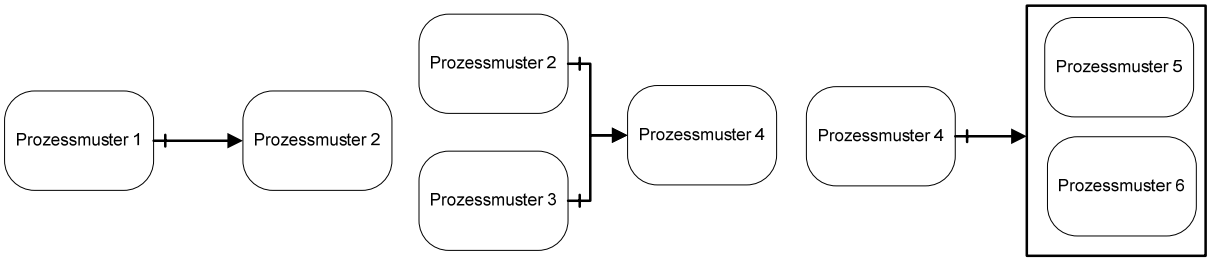



**Abbildung 31: Beispiel Sicht "Sequence-Beziehung" in Anlehnung an [Hage05, 151]**

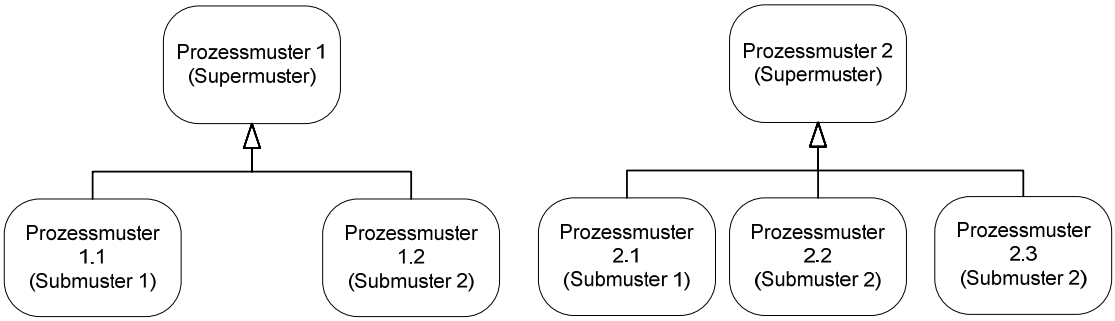



**Abbildung 32: Beispiel Sicht "Refinement-Beziehung" in Anlehnung an [Hage05, 152]**

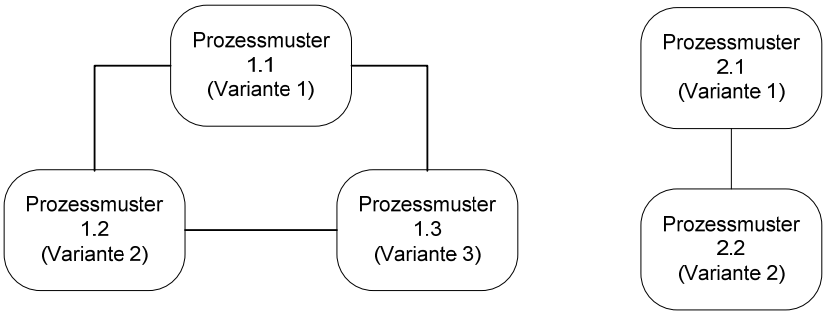



**Abbildung 33: Beispiel Sicht "Processvariance-Beziehung" in Anlehnung an [Hage05, 152]**

Im Prozessmusterkatalog werden auch die Objekt- und Ereignisstruktur, der verwendeten Objekte und Ereignisse angeführt. Dabei wird die Aggregation und Generalisierung der Objekte und Ereignisse verwendet. Die Aggregation dient dazu, die Übersichtlichkeit durch die Objekt- und Ereigniskomposition zu erhöhen. Durch Generalisierung bzw. Spezialisierung können auch abstraktere bzw. konkretere Ausprägungen der Objekte und Ereignisse als Kontext der Prozessmuster verwendet werden (z.B. durch das abstrakte Objekt „Review

Object“ kann das Prozessmuster „Review Requirements“ für jede konkrete Ausprägung dieses Objekts verwendet werden). [Hage05, 153f] Wie bereits erwähnt, hat die Objekt- und Ereigniskomposition Einfluss auf die Prozessmusterbeziehungen. [Hage05, 62ff]

Für die Untersuchung der reaktiven Wiederverwendung wird ein Prozessmusterkatalog aus den Prozessmodellen aus Abschnitt 1.4 in Abschnitt 3.4.1 in Anlehnung an die Beispiele aus Hagen erstellt (vgl. [Hage05, 147ff]). Das Rahmenbeispiel wird für die Untersuchung aus Gründen der Komplexität einfach gehalten. Daher wird auf die Darstellung der verschiedenen Sichten verzichtet.

Auf eine separate Darstellung der Objekt- und Ereignisstruktur wird in der Untersuchung ebenfalls verzichtet, da nur Ereignisse als initialer und resultierender Kontext verwendet werden.

## *3.4 Wiederverwendung anhand des Rahmenbeispiels*

Die Anwendung von Prozessmuster wird unter dem Begriff *Process Pattern Management* zusammengefasst, der folgende Tätigkeiten beinhaltet [Hage05, 25]:

- Dokumentation von Prozessmuster
- Identifikation und Spezifikation eines Problems
- Suche nach Prozessmuster
- Auswahl von Prozessmuster
- Anwendung, Einführung, Einsatz und Verbesserung von Prozessmuster
- Bereitstellung von Werkzeugen zur Unterstützung dieser Aufgaben

Die Process Pattern Workbench, ein Werkzeug im Stadium eines Prototyps, übernimmt die Aufgaben der Sammlung und Aufbereitung von Prozessmustern und überprüft darüber hinaus die Prozessmuster und ihre Beziehungen auf ihre syntaktische Richtigkeit. Jedoch muss unbedingt entschieden werden, nach welchem Schema die Prozessmuster und Probleme zu beschreiben sind. [Hage05, 201] Damit kommt zum Ausdruck, dass die vorher beschriebenen Schemata ein Vorschlag sind.

Im Rahmen der Untersuchung wird auf die Dokumentation von Prozessmuster, die Identifikation und Spezifikation eines Problems, die Suche nach Prozessmuster, die Auswahl von Prozessmuster und die Anwendung von Prozessmuster eingegangen. Die Aufgaben der Einführung, den Einsatz und die Verbesserung von Prozessmuster und der Bereitstellung von Werkzeugen zur Unterstützung dieser Aufgaben werden vernachlässigt, da diese nicht Gegenstand der reaktiven Wiederverwendung und daher nicht Gegenstand dieser Untersuchung sind. Die Aufgaben der Überprüfung werden zwar nicht explizit genannt, sind jedoch implizit vorhanden. Nachfolgender Tabelle ist eine Gegenüberstellung der Aufgaben nach Hagen und der verwendeten Einteilung zu entnehmen:

| Einteilung | Hagen |
|---|---|
| Integration von externen Prozesslösungen | • Dokumentation von Prozessmuster |
| Suche nach geeigneten Prozesslösungen | • Identifikation und Spezifikation eines Problems<br>• Suche nach Prozessmuster<br>• Auswahl von Prozessmuster |
| Anwendung von gefundenen Prozesslösungen | • Anwendung von Prozessmuster |
| Überprüfung der angepassten Prozesslösungen | • Implizit vorhanden |
| Integration neuer Prozesslösungen in die Wissensbasis | • Dokumentation von Prozessmuster |

**Tabelle 5: Zuordnung der gemeinsamen Konzepte und Aufgaben nach Hagen**

In diesem Abschnitt wird nun das Vorgehensmodell anhand der in Abschnitt 2.2.2.1 eingeführten Einteilung zur reaktiven Wiederverwendung vorgestellt: *Integration von externen Prozesslösungen* (Abschnitt 3.4.1), *Suche nach geeigneten Prozesslösungen* (Abschnitt 3.4.2), *Anwendung von gefundenen Prozesslösungen* (Abschnitt 3.4.3), *Überprüfung von angepassten Prozesslösungen* (Abschnitt 3.4.4) und *Integration neuer Prozesslösungen in die Wissensbasis* (Abschnitt 3.4.5) erklärt.

### 3.4.1 Integration von externen Prozesslösungen

Um Prozessmuster für die reaktive Wiederverwendung wiederverwendbar zu machen, müssen die Prozessmuster definiert und modelliert werden. Hagen teilt diese Aufgaben den Rollen des Pattern Designers und Pattern Users zu. Der Pattern Designer übernimmt die Aufgaben, die Pattern-Idee zu entwickeln, zu bewerten und anschließend anhand eines einheitlichen Schemas zu dokumentieren (vgl. Prozessmuster- und Problemschemata). Hagen weist die Aufgaben noch genauer zu, jedoch sind diese im Rahmen dieser Diplomarbeit nicht von Bedeutung. [Hage05, 204f]

In diesem Abschnitt wird der Prozessmusterkatalog als Wissensbasis für die anschließende reaktive Wiederverwendung erstellt. Ausgangsbasis zur Integration von externen Prozesslösungen bilden die dieser Untersuchung zugrunde gelegten eEPK Prozessmodelle aus Abschnitt 1.4. Zu diesen Prozessmodellen werden Prozessmuster anhand eines vereinfachten Schemas extrahiert (vgl. Abbildung 34). Es wird für jedes Prozessmodell ein Prozessmuster definiert, das wiederum in mehrere Teilmodelle aufgeteilt wird. Für jedes Teilmodell wird ebenfalls ein Prozessmuster dokumentiert. Zu jedem Prozessmuster werden die Probleme mit einer Frage beschrieben, sowie das Startereignis bzw. das Endereignis des (Teil-)Prozesses als initialer bzw. als resultierender Kontext angegeben. Um den Rahmen der Untersuchung nicht zu sprengen, wird auf das Hinzuziehen von Objekten verzichtet. Anschließend werden die lösenden Prozessmuster angeführt. Dies wird mit Hilfe des unten angeführten Schemas dargestellt.

*Px*… Prozessmuster
*P*… Problem
*C*… Kontext…
Initialer Kontext
Resultierender Kontext
*S*… Lösung

**Abbildung 34: Schema**

Ausgehend von den erstellten Schemata werden zuerst Problemdiagramme erstellt. Diese beinhalten den initialen und den resultierenden Kontext und das Problem sowie die lösenden Prozessmuster (vgl. z.B. [Hage05, 154f]). Anschließend werden Prozessmusterdiagramme erstellt, die das zu lösende Problem, ein Beziehungsdiagramm und ein Prozessdiagramm beinhalten (vgl. z.B. [Hage05, 156]). Das Beziehungsdiagramm stellt alle mit dem Prozessmuster *„in Beziehung stehenden Prozessmuster“* dar [Hage05, 143], wobei alle vorher erklärten Beziehungstypen verwendet werden (Sequence-, Use-, Refinement- und Processvariance-Beziehung) (vgl. Abschnitt 3.2). Das Prozessdiagramm repräsentiert die Lösung selbst [Hage05, 68].

Zu jedem Prozessmodell aus Abschnitt 1.4 werden schlussendlich die Beziehungen zwischen den Prozessmustern identifiziert, und so der Prozessmusterkatalog erstellt.

***Beispiel 3.4.1.1***

*Zu Beginn werden zu dem Prozessmodell model_100 die Ausprägungen in das oben erwähnte Schema eingetragen, womit dieses das erste Prozessmuster P0 ergibt.*

*P0…*
*Name: Kreditantrag bearbeiten*
*P… „Wie wird der Kreditantrag bearbeitet?“*
*C… Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_0*

*Das Prozessmodell model_100 kann in mehrere Teilmodelle aufgesplittet werden, und es können weitere Prozessmuster definiert werden. Damit wird eine Hierarchie geschaffen, da folgende Prozessmuster von P0 genutzt werden (vgl. Prozessmusterbeziehungen).*

*P0.1…*
*Name: Kreditrisiko ermitteln*
*P… „Wie wird das Kreditrisiko ermittelt?“*
*C… Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditrisiko ermittelt“*
*S… Prozessmodell_0.1*

*P0.2…*

*Name: Kreditantrag ablehnen*
*P… „Wie wird der Kreditantrag abgelehnt?“*
*C… Initialer Kontext… „Kreditrisiko hoch“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_0.2*

*P0.3…*

*Name: Kreditantrag bewilligen*
*P… „Wie wird der Kreditantrag bewilligt?“*
*C… Initialer Kontext… „Kreditrisiko niedrig“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_0.3*

*Die gleiche Vorgehensweise wird auf das Prozessmodell model_101 angewendet. Es wird ebenfalls in das gleiche Schema übertragen, um den Prozessmusterkatalog vollständig mit allen Prozessmusterbeziehungen zu erstellen. Da es den gleichen Kontext besitzt wie das Prozessmodell model_100, ist P1 eine Prozessvariante zu P0.*

*P1…*

*Name: Kreditantrag bearbeiten*
*P… „Wie wird der Kreditantrag bearbeitet?“*
*C… Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_1*

*Dieses Prozessmodell (model_101) kann ebenfalls in mehrere Teilmodelle aufgeteilt werden, sodass weitere Prozessmuster dazu dokumentiert werden.*

*P1.1…*

*Name: Kreditrisiko ermitteln*
*P… „Wie wird das Kreditrisiko ermittelt?“*
*C… Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditrisiko ermittelt“*
*S… Prozessmodell_1.1*

*P1.2…*

*Name: Kreditantrag ablehnen*
*P… „Wie wird der Kreditantrag abgelehnt?“*
*C… Initialer Kontext… „Kreditrisiko hoch“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_1.2*

*P1.3…*

*Name: Kreditantrag bewilligen*
*P… „Wie wird der Kreditantrag bewilligt?“*
*C… Initialer Kontext… „Kreditrisiko niedrig“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_1.3*

*Somit ergibt sich folgendes Problemdiagramm (Abbildung 35) für das Problem „Wie wird der Kreditantrag bearbeitet?“, das von den Prozessmustern P0 „Kreditantrag bearbeiten“ und P1 „Kreditantrag bearbeiten“ gelöst wird. Diese beiden Prozessmuster stellen Prozessvarianten dar, da beide den gleichen Kontext haben.*

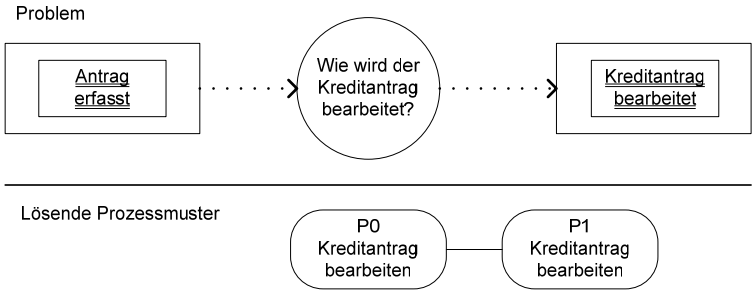



**Abbildung 35: Problemdiagramm "Wie wird der Kreditantrag bearbeitet?"**

*Wie aus dem oben angeführten Schema ersichtlich ist, besteht der initiale Kontext aus dem Ereignis „Antrag erfasst“ (zur linken Seite des Problems) und der resultierende Kontext aus dem Ereignis „Kreditantrag bearbeitet“ (zur rechten Seite des Problems). Die lösenden Prozessmuster sind die Prozessmuster P0 „Kreditantrag bearbeiten“ und P1 „Kreditantrag bearbeiten“.*

*Das Prozessmusterdiagramm zu dem Prozessmuster P0 „Kreditantrag bearbeiten“ ist in Abbildung 36 dargestellt. Zu Beginn wird das zu lösende Problem angeführt. Anschließend wird das Beziehungsdiagramm dargestellt, das alle Prozessmuster beinhaltet, die in Beziehung zu dem Prozessmuster stehen. Hier ist zu erkennen, dass das Prozessmuster P0 jeweils eine Use-Beziehung zu den Prozessmustern P0.1 „Kreditrisiko ermitteln“, P0.2 „Kreditantrag ablehnen“ und P0.3 „Kreditantrag bewilligen“ aufweist. Dies bedeutet, dass das Prozessmuster P0 das Kompositmuster und die Prozessmuster P0.1, P0.2 und P0.3 Komponentenmuster laut der Definition von Hagen sind (vgl. Prozessmusterbeziehungen). Der dritte Teil der Abbildung stellt das lösende Prozessdiagramm dar, das in dieser Arbeit wie bereits erwähnt als eEPK anstatt durch ein UML-Aktivitätsdiagramm modelliert ist.*

*In Abbildung 37 wird das Prozessmusterdiagramm zu Prozessmuster P1 „Kreditantrag bearbeiten“ nach der gleichen Einteilung dargestellt.*

*Dass es sich hier um Prozessvarianten handelt, kann man daran erkennen, dass zu Beginn das gleiche Problem angeführt ist. Im Beziehungsdiagramm werden hier ebenfalls alle Prozessmuster angeführt, die in Beziehung zu dem Prozessmuster P1 stehen, also P1.1 „Kreditrisiko ermitteln“, P1.2 „Kreditantrag ablehnen“ und P1.3 „Kreditantrag bewilligen“. Auch hier ist zu erkennen, dass das Prozessmuster P1 das Kompositmuster ist und P1.1, P1.2 und P1.3 die Komponentenmuster.*

*Im unteren Teil der Abbildung ist als lösendes Prozessdiagramm das Prozessmodell model_101 angeführt.*

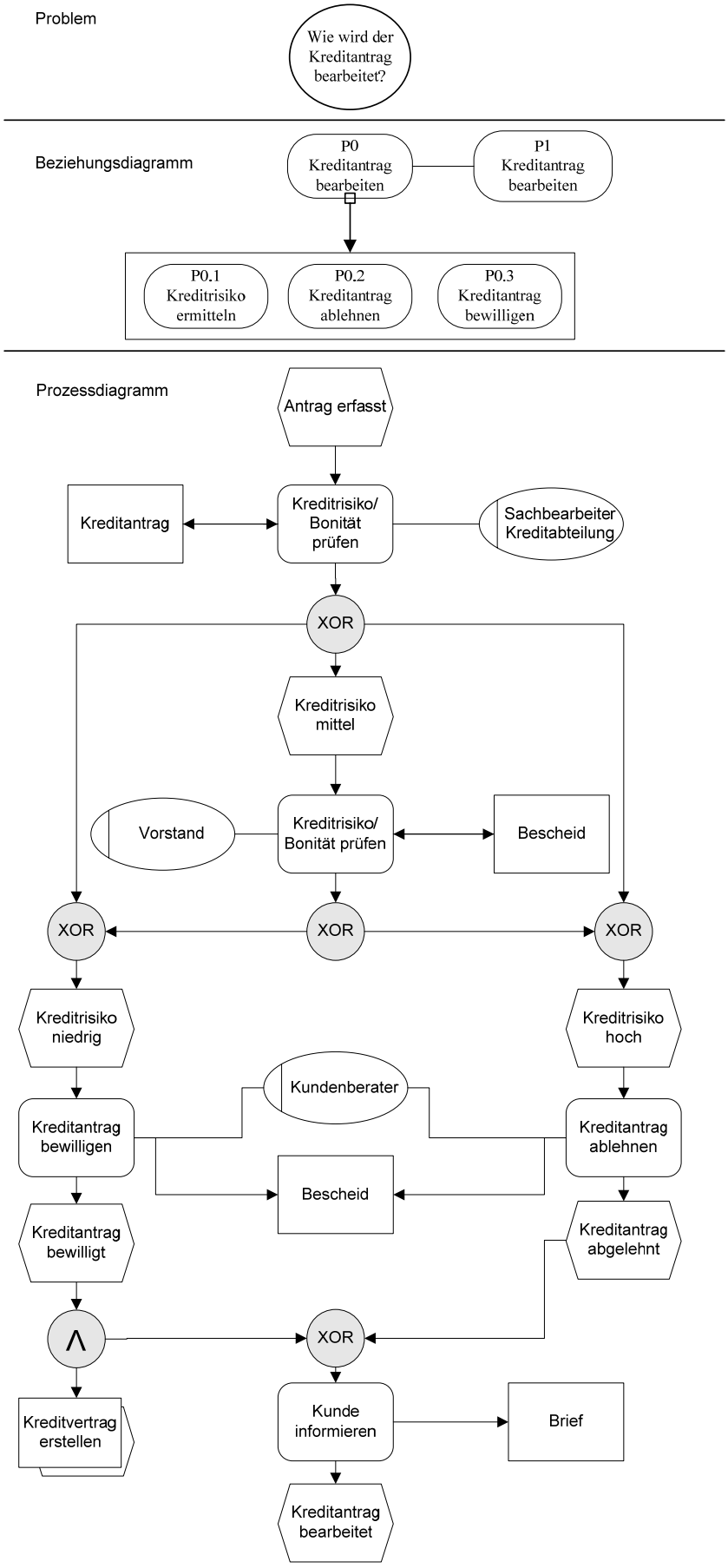



**Abbildung 36: Prozessmusterdiagramm P0 "Kreditantrag bearbeiten"**

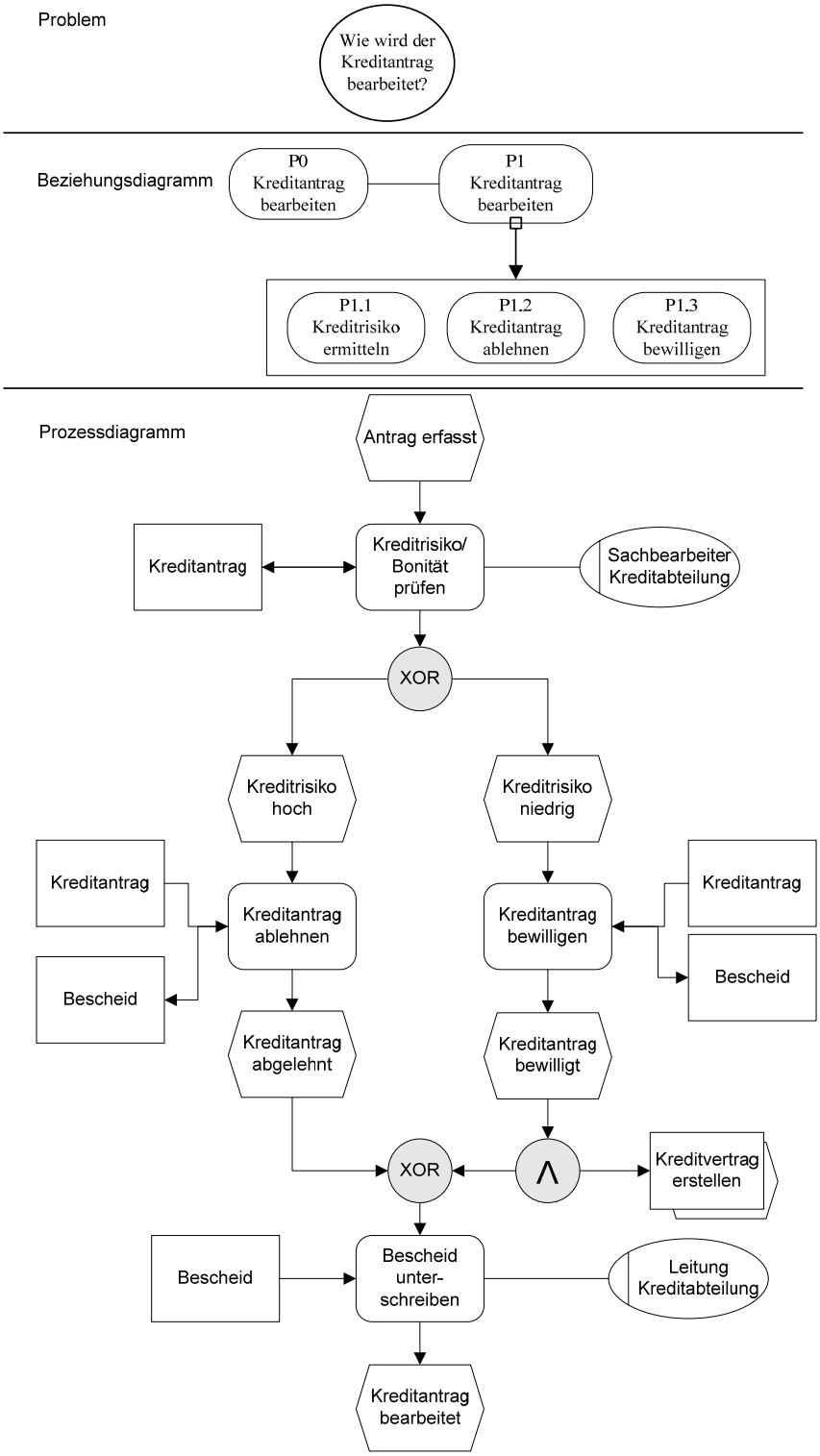



**Abbildung 37: Prozessmusterdiagramm P1 "Kreditantrag bearbeiten"**

*Für das Problem „Wie wird das Kreditrisiko ermittelt?“ ergibt sich das in Abbildung 38 dargestellte Problemdiagramm. Dieses Problem wird von den Prozessmustern P0.1 und P1.1 gelöst.*

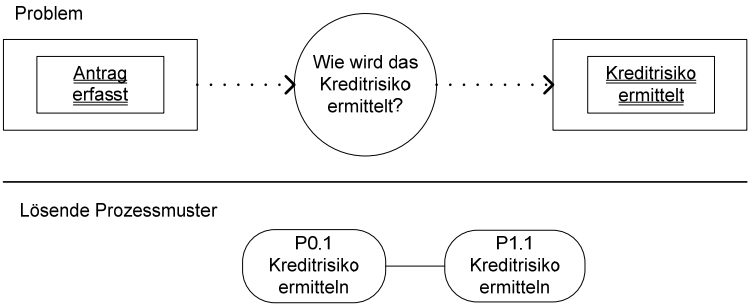



**Abbildung 38: Problemdiagramm "Wie wird das Kreditrisiko ermittelt?"**

*In dem oben angeführten Diagramm ist als initialer Kontext zu dem Problem „Wie wird das Kreditrisiko ermittelt?“ das Ereignis „Antrag erfasst“ (zur linken Seite des Problems) und der resultierende Kontext aus dem Ereignis „Kreditrisiko ermittelt“ (zur rechten Seite des Problems) angeführt. Die lösenden Prozessmuster sind die Prozessmuster P0.1 und P1.1 „Kreditrisiko ermitteln“. Auch hier ist eine Processvariance-Beziehung zwischen den beiden Prozessmustern angeführt, da beide das gleiche Problem lösen.*

*Das Prozessmusterdiagramm zu dem Prozessmuster P0.1 „Kreditrisiko ermitteln“ ist in Abbildung 39 dargestellt. Auch hier wird zuerst das zu lösende Problem angeführt, anschließend das Beziehungsdiagramm mit allen Prozessmustern, die in Beziehung zu dem Prozessmuster P0.1 stehen. Hier ist zu erkennen, dass das Prozessmuster P0.1 zusätzlich zu der Use-Beziehung zu dem Prozessmuster P0, eine Sequence-Beziehung zu den Prozessmustern P0.2 „Kreditantrag ablehnen“ und P0.3 „Kreditantrag bewilligen“ sowie eine Processvariance-Beziehung zu P1.1 aufweist. Durch die Sequence-Beziehung wird ausgesagt, dass P0.2 und P0.3 nach P0.1 ausgeführt werden.*

*Im unteren Teil der Abbildung ist das lösende Prozessdiagramm, das einen Teilprozess des Prozessmodells model_100 darstellt.*

*Das Prozessmusterdiagramm zu dem Prozessmuster P1.1 „Kreditrisiko ermitteln“, das das gleiche Problem löst, ist in Abbildung 40 dargestellt. Im Beziehungsdiagramm besteht eine Use-Beziehung zu dem Prozessmuster P1 und eine Sequence-Beziehung zu den Prozessmustern P1.2 „Kreditantrag ablehnen“ und P1.3 „Kreditantrag bewilligen“ und analog zu vorher eine Processvariance-Beziehung zu P0.1 aufweist.*

*Das lösende Prozessdiagramm ist ebenfalls im unteren Teil der Abbildung. Es ist ein Teil des Prozesses des Prozessmodells model_101.*

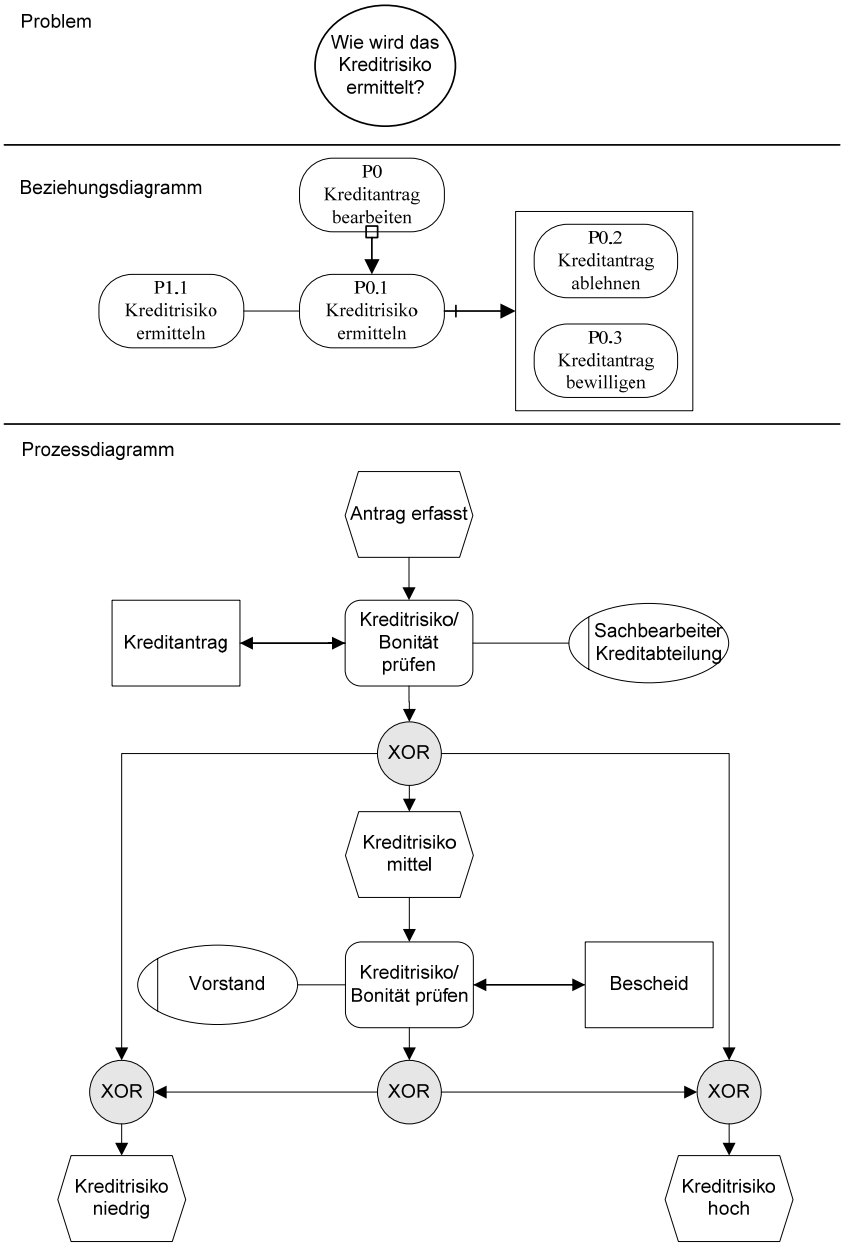



**Abbildung 39: Prozessmusterdiagramm P0.1 "Kreditrisiko ermitteln"**

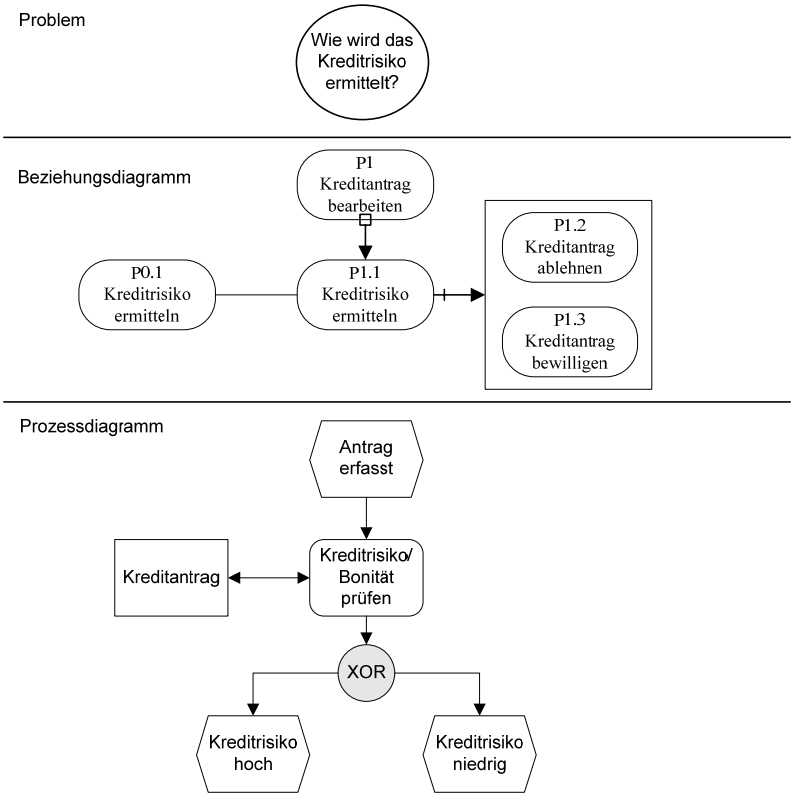



**Abbildung 40: Prozessmusterdiagramm P1.1 "Kreditrisiko ermitteln"**

*Für das Problem „Wie wird das Kreditrisiko abgelehnt?“ ergibt sich das in Abbildung 41 dargestellte Problemdiagramm. Dieses Problem wird von den Prozessmustern P0.2 und P1.2 „Kreditantrag ablehnen“ gelöst.*

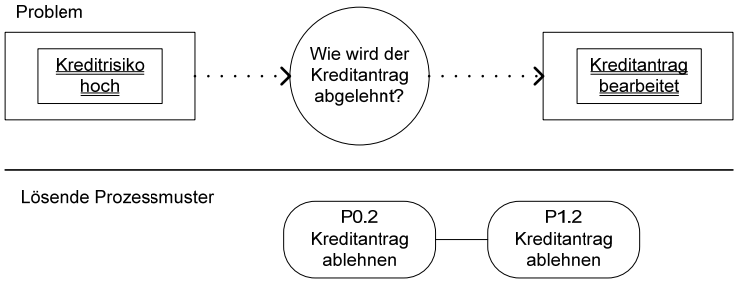



**Abbildung 41: Problem "Kreditantrag ablehnen"**

*Der initiale Kontext zu diesem Problem („Wie wird der Kreditantrag abgelehnt?“) ist das Ereignis „Kreditrisiko hoch“ (zur linken Seite des Problems) und der resultierende Kontext ist das Ereignis „Kreditantrag bearbeitet“ (zur rechten Seite des Problems). Die lösenden Prozessmuster sind P0.2 und P1.2 „Kreditantrag ablehnen“, die mit einer Processvariance-Beziehung verknüpft sind.*

*Analog zu den vorherigen Prozessmusterdiagrammen werden zu diesem Problem ebenfalls zwei im Katalog angeführt. Das Prozessmusterdiagramm zu dem Prozessmuster P0.2 „Kreditantrag ablehnen“ ist in Abbildung 42 dargestellt. Es weist im Beziehungsdiagramm eine Use-Beziehung und eine Processvariance-Beziehung auf. Es enthält im unteren Teil einen Teil des Gesamtprozesses des Prozessmodells model_100.*

*Das Prozessmusterdiagramm zu P1.2 (Abbildung 43) weist im Beziehungsdiagramm ebenfalls eine Use-Beziehung und eine Processvariance-Beziehung auf. Im unteren Teil enthält es einen Ausschnitt des Gesamtprozesses zu dem Prozessmodell model_101.*

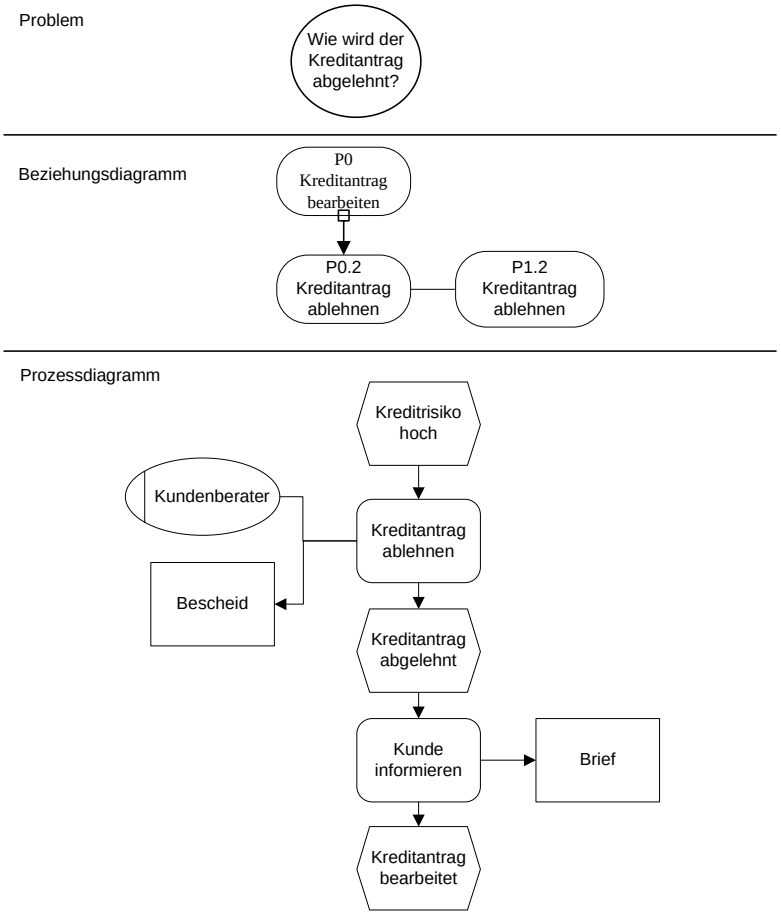



**Abbildung 42: Prozessmusterdiagramm P0.2 "Kreditantrag ablehnen"**

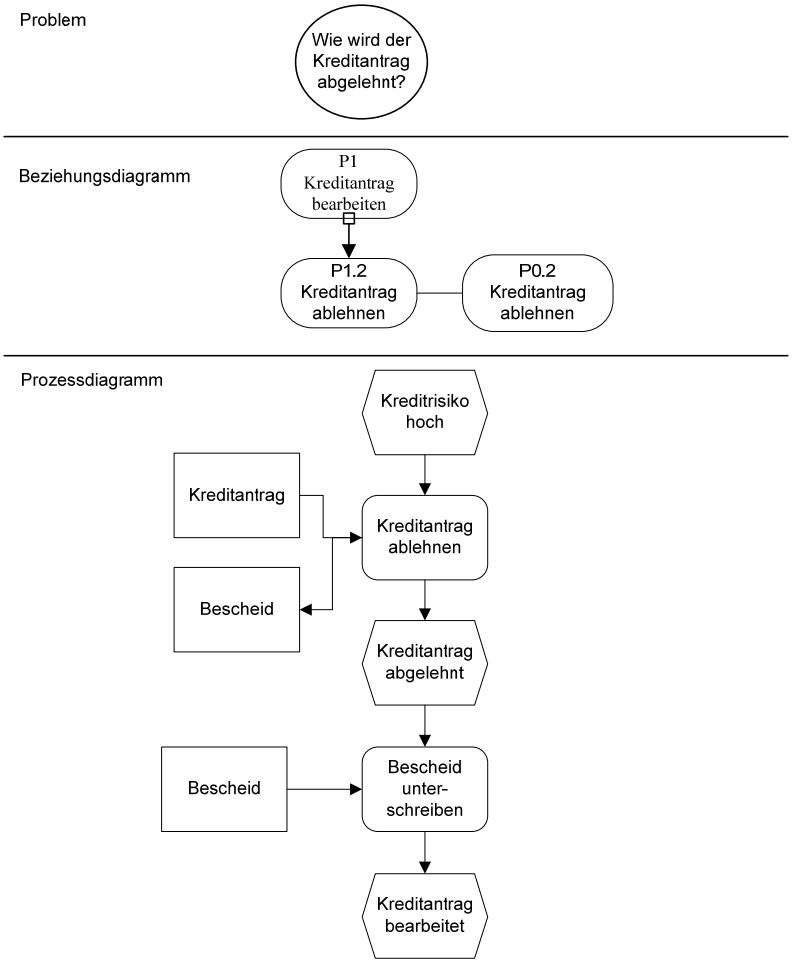



**Abbildung 43: Prozessmusterdiagramm P1.2 "Kreditantrag ablehnen"**

*Für das Problem „Wie wird der Kreditantrag bewilligt?“ ergibt sich das in Abbildung 44 dargestellte Problemdiagramm. Dieses Problem wird von den Prozessmustern P0.3 und P1.3 „Kreditantrag ablehnen“ gelöst.*

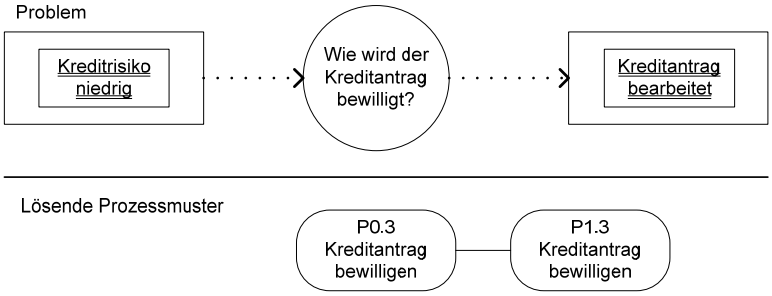



**Abbildung 44: Problem "Wie wird der Kreditantrag bewilligt?"**

*Der initiale Kontext zu diesem Problem („Wie wird der Kreditantrag bewilligt?“) ist das Ereignis „Kreditrisiko niedrig“ (zur linken Seite des Problems) und der resultierende Kontext ist das Ereignis „Kreditantrag bearbeitet“ (zur rechten Seite des Problems). Die lösenden Prozessmuster sind P0.3 und P1.3 „Kreditantrag ablehnen“, die mit einer Processvariance-Beziehung verknüpft sind, da sie den gleichen Kontext (initial und resultierend) aufweisen.*

*Auch zu diesem Problem wird für jede Prozessvariante ein Prozessmusterdiagramm im Katalog angeführt. Das Prozessmusterdiagramm zu dem Prozessmuster P0.3*

*„Kreditantrag bewilligen“ ist in Abbildung 45 und enthält im unteren Teil einen Teil des Gesamtprozesses des Prozessmodells model_100. Jenes zu P1.3 in Abbildung 46 enthält im unteren Teil ebenfalls einen Ausschnitt eines Gesamtprozesses, und zwar zu dem Prozessmodell model_101.*

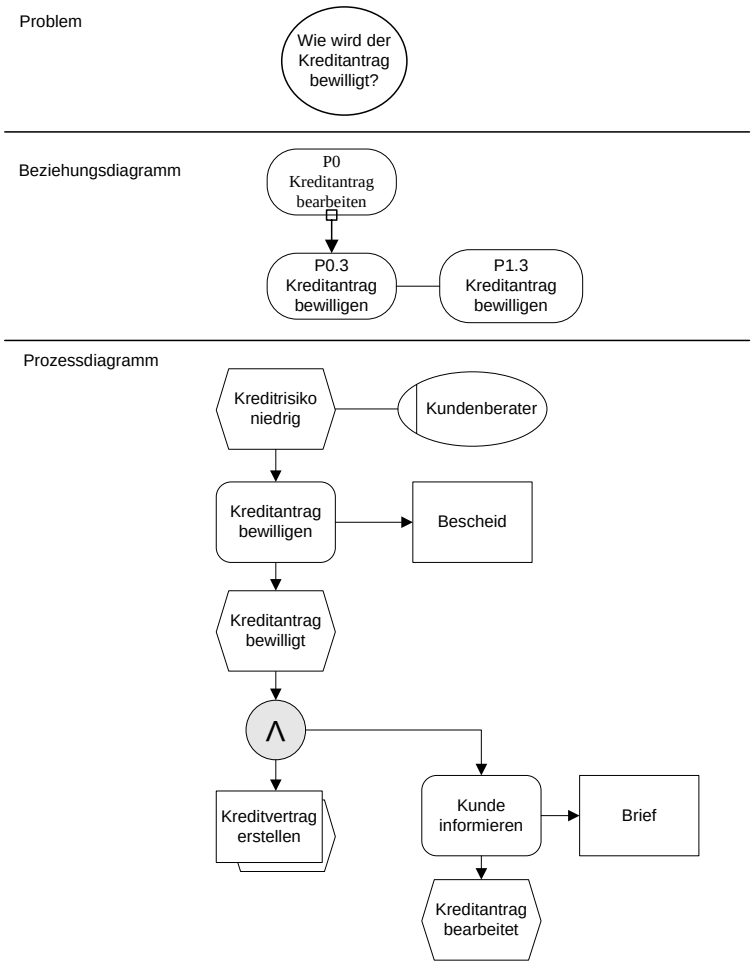



**Abbildung 45: Prozessmusterdiagramm P0.3 "Kreditantrag bewilligen"**

*Für den Prozesspfad „Kreditvertrag erstellen“ ist kein Prozess in Abschnitt 1.4 angeführt, daher gibt es in weiterer Folge auch kein weiteres Prozessmuster. Dieses Prozessmuster würde analog zu dem Prozessmuster P3.2 „Kredithöhe für Privatkredit ermitteln“ für den Prozesspfad „Max. Kredithöhe“ und dem Prozessmuster P3.3 „Kredithöhe für Beleihungsgrenze ermitteln“ für den Prozesspfad „Beleihungsgrenze“ erstellt werden (vgl. weiter unten), indem ein Prozessmuster den Prozesspfad enthält und eines oder mehrere Prozessmuster die detaillierten Prozessschritte enthält und über jeweils eine Use-Beziehung verknüpft sind.*

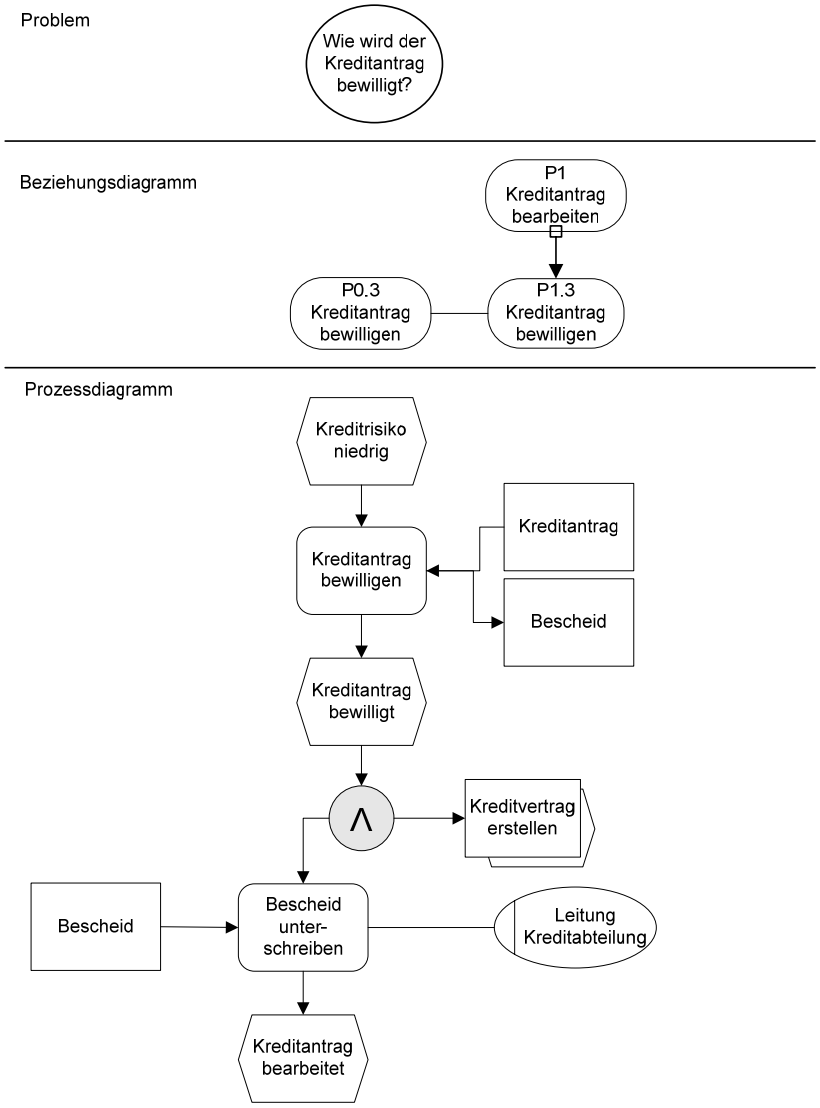



**Abbildung 46: Prozessmusterdiagramm P1.3 "Kreditantrag bewilligen"**

*Das gesamte Beziehungsdiagramm im Prozessmusterkatalog zu den Prozessmodellen model_100 und model_101 ist in folgender Abbildung zu sehen.*

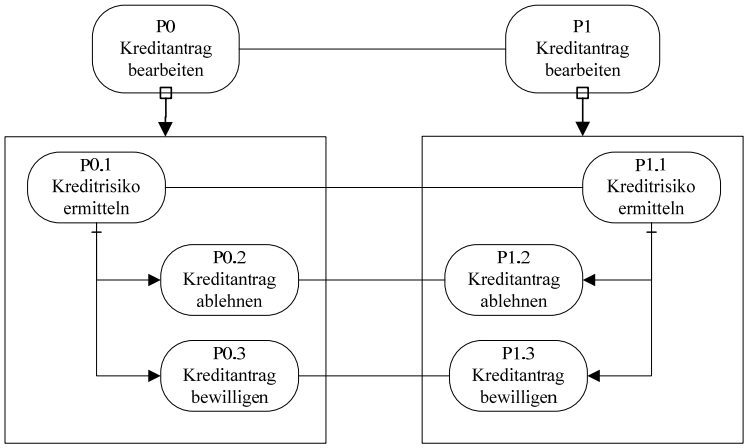



**Abbildung 47: Beziehungsdiagramm model_100 und model_101**

*Da zu den Aufgaben des Pattern Designers auch die Entwicklung neuer Prozessmuster gehört, kann aus diesen beiden Prozessmodellen durch Anwendung der Prozessmuster*

*ein neues Prozessmodell erstellt werden. Es können für ein neues Prozessmuster P2 „Kreditantrag bearbeiten“ z.B. die Prozessmuster P1.1, P0.2 und P1.3 verwendet werden und über Use-Beziehungen verknüpft werden (Abbildung 48). Dies ist eine Möglichkeit. Es können auch andere Prozessmuster kombiniert werden, solange die Regeln der Syntax eingehalten werden (Übereinstimmung der Kontexte).*

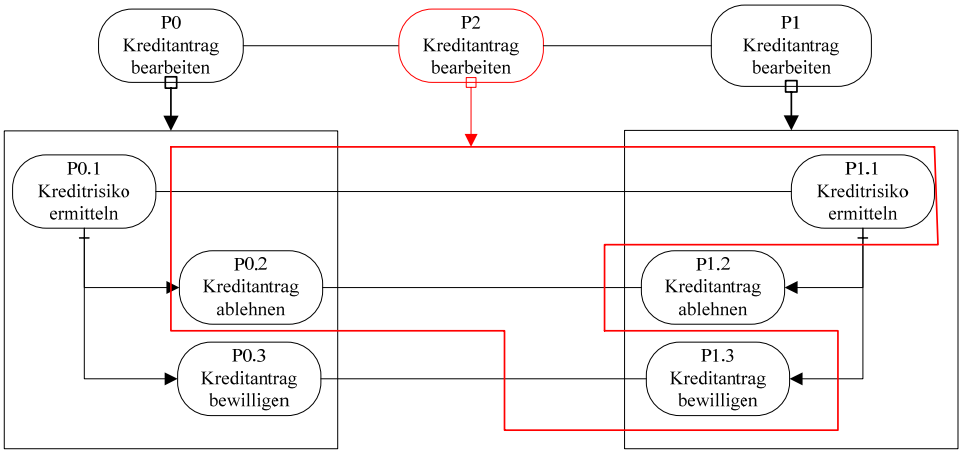



**Abbildung 48: Neues Beziehungsdiagramm**

*Somit wird das Problemdiagramm wie in Abbildung 49 dargestellt verändert. Es kommt das neue Prozessmuster als Prozessvariante hinzu und wird durch eine Processvariance-Beziehung zu P0 und P1 verknüpft, da es das gleiche Problem löst und den gleichen Kontext besitzt (vgl. den Kontext des Prozessmusters mit den Kontexten von P0 und P1 auf Seite 62).*

*P2…*

*Name: Kreditantrag bearbeiten*
*P… „Wie wird der Kreditantrag bearbeitet?“*
*C… Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*
*S… Prozessmodell_2*

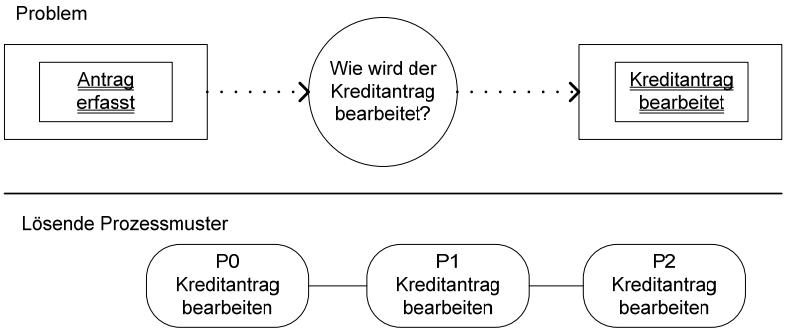



**Abbildung 49: Problemdiagramm "Wie wird der Kreditantrag bearbeitet?"**

*Prozessmusterdiagramm für das Prozessmuster P2 „Kreditantrag bearbeiten“ ist in Abbildung 50 zu sehen. Das Prozessmuster P2 nutzt die Prozessmuster P1.1, P0.2 und P1.3. Daher wird zusätzlich zu den Processvariance-Beziehungen eine Use-Beziehung zu den genutzten Prozessmustern im Beziehungsdiagramm angegeben. Im unteren Teil (Prozessdiagramm) werden die einzelnen Prozessschritte angeführt.*

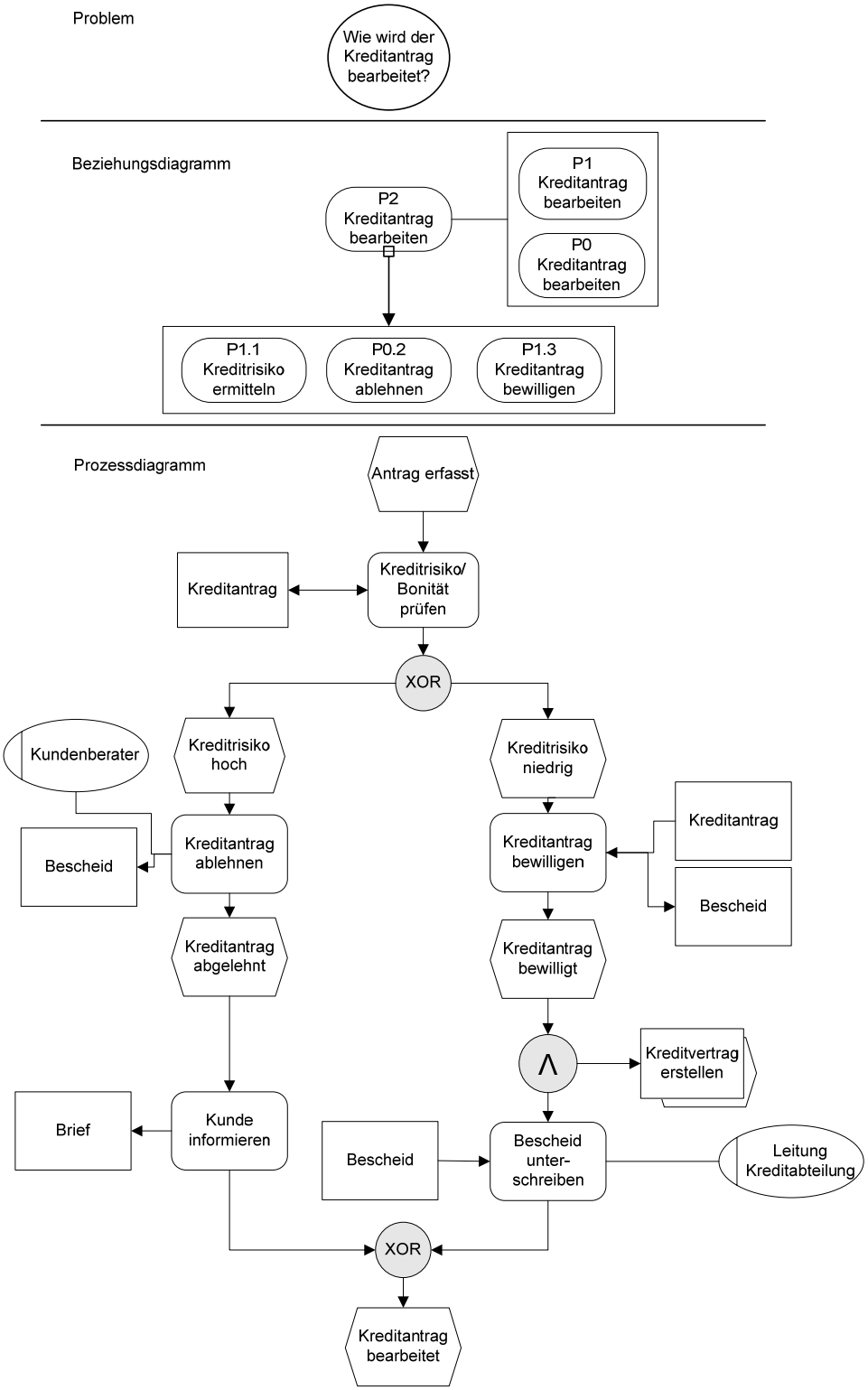



**Abbildung 50: Prozessmusterdiagramm P2 „Kreditantrag bearbeiten“**

*Nun müssen für die restlichen Prozessmodelle in der Wissensbasis (model_102, model_1021 und model_1022) ebenfalls Prozessmuster dokumentiert werden. Dafür wird wiederum das Tripel Problem, Kontext und Lösung identifiziert und in der gleichen Einteilung angeführt.*

*P3…*

*Name: Kredithöhe für Hypothekar- oder Privatkredite ermitteln*
*P… „Wie wird die Kredithöhe für Hypothekar- oder Privatkredite ermittelt?“*
*C… Initialer Kontext… „Kreditwunsch ist vorgetragen“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3*

*P3.1…*

*Name: Kredit beantragen*
*P… „Wie wird der Kredit beantragt?“*
*C… Initialer Kontext… „Kreditwunsch ist vorgetragen“*
*Resultierender Kontext… „Kredit ist beantragt“*
*S… Prozessmodell_3.1*

*P3.2…*

*Name: Kredithöhe für Privatkredit ermitteln*
*P… „Wie wird die Kredithöhe für den Privatkredit ermittelt?“*
*C… Initialer Kontext… „Privatkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3.2*

*P3.2.1…*

*Name: Kredithöhe für Privatkredit ermitteln (detailliert)*
*P… „Wie wird die Kredithöhe für den Privatkredit ermittelt (detailliert)?“*
*C… Initialer Kontext… „Privatkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3.2*

*P3.2.1.1…*

*Name: Unterlagen beschaffen*
*P… „Wie werden die Unterlagen für den Privatkredit beschafft?“*
*C… Initialer Kontext… „Privatkredit ist beantragt“*
*Resultierender Kontext… „Unterlagen verfügbar“*
*S… Prozessmodell_3.2.1.1*

*P3.2.1.2…*

*Name: Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln*
*P… „Wie wird die Kredithöhe für den Privatkredit mit den Unterlagen ermittelt?“*
*C… Initialer Kontext… „Unterlagen sind verfügbar“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3.2.1.2*

*P3.3…*

*Name: Kredithöhe für Hypothekarkredit ermitteln*
*P… „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“*
*C… Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3.3*

*P3.3.1…*

*Name: Kredithöhe für Hypothekarkredit ermitteln (detailliert)*

*P… „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“*

*C… Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*

*S… Prozessmodell_3.3.1*

*P3.3.1.1…*

*Name: Beleihungsgrenze für Hypothekarkredit ermitteln*

*P… „Wie wird die Beleihungsgrenze für den Hypothekarkredit ermittelt?“*

*C… Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Beleihungsgrenze ermittelt“ und „Einkommensnachweis verfügbar“*

*S… Prozessmodell_3.3.1.1*

*P3.3.1.2…*

*Name: Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln*

*P… „Wie wird die Kredithöhe für den Hypothekarkredit mit Unterlagen ermittelt?“*

*C… Initialer Kontext… „Beleihungsgrenze ermittelt“ und „Einkommensnachweis verfügbar“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*

*S… Prozessmodell_3.3.1.2*

*Somit ergibt sich folgendes Problemdiagramm (Abbildung 51) für das Problem „Wie wird die Kredithöhe für Hypothekar- oder Privatkredite ermittelt?“, das von dem Prozessmuster P3 „Kredithöhe für Hypothekar- oder Privatkredite ermitteln“ gelöst wird.*

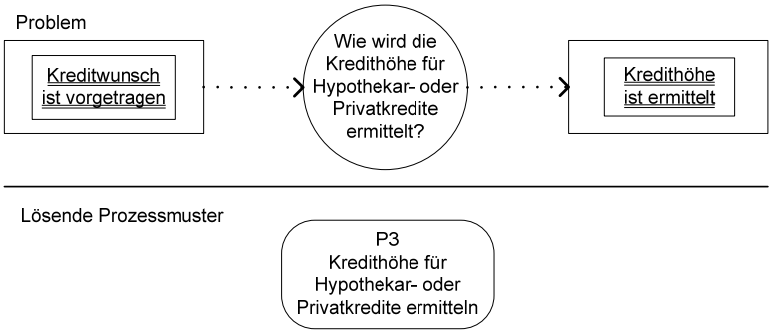



**Abbildung 51: Problemdiagramm "Wie wird die Kredithöhe für Hypothekar- oder Privatkredite ermittelt?“**

*Wie aus diesem Schema ersichtlich ist, besteht der initiale Kontext aus dem Ereignis „Kreditwunsch ist vorgetragen“ (zur linken Seite des Problems) und der resultierende Kontext aus dem Ereignis „Kredithöhe ist ermittelt“ (zur rechten Seite des Problems). Das lösende Prozessmuster ist das Prozessmuster P3 „Kredithöhe für Hypothekar- oder Privatkredite ermitteln“.*

*Das Prozessmusterdiagramm zu dem Prozessmuster P3 „Kredithöhe für Hypothekar- oder Privatkredite ermitteln“ ist in Abbildung 52 dargestellt.*

*Dieses Prozessmuster weist in dem Beziehungsdiagramm eine Use-Beziehung zu den Prozessmustern P3.1 „Kredit beantragen“, P3.2 „Kredithöhe für Privatkredit ermitteln“ und P3.3 „Kredithöhe für Hypothekarkredit ermitteln“ auf, da diese von dem Prozessmuster verwendet werden. Im Prozessdiagramm ist das Prozessmodell model_102 angeführt.*

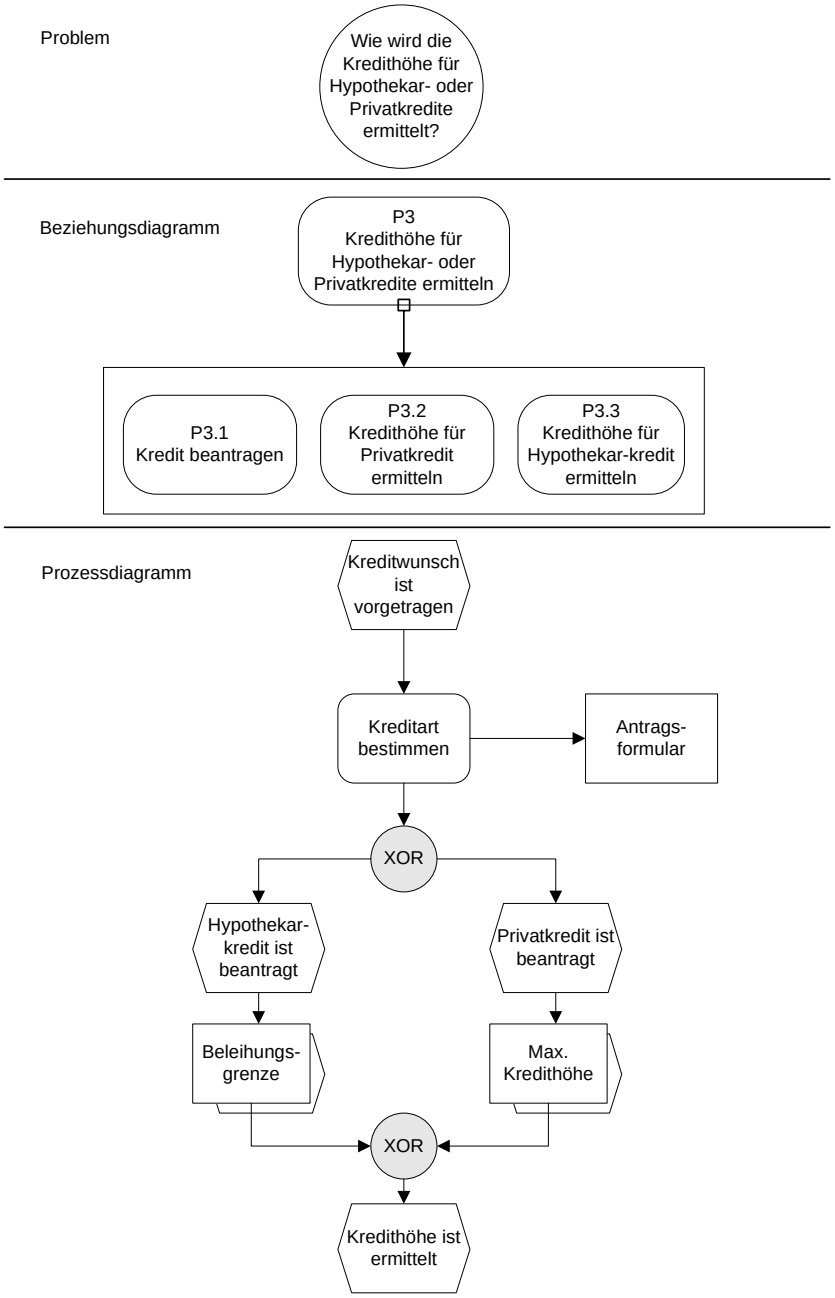



**Abbildung 52: Prozessmusterdiagramm P3 "Kredithöhe für Hypothekar- oder Privatkredite ermitteln"**

*Das Problem „Wie wird der Kredit beantragt?“, das von dem Prozessmuster P3.1 „Kredit beantragen“ gelöst wird, wird in Abbildung 53 dargestellt.*

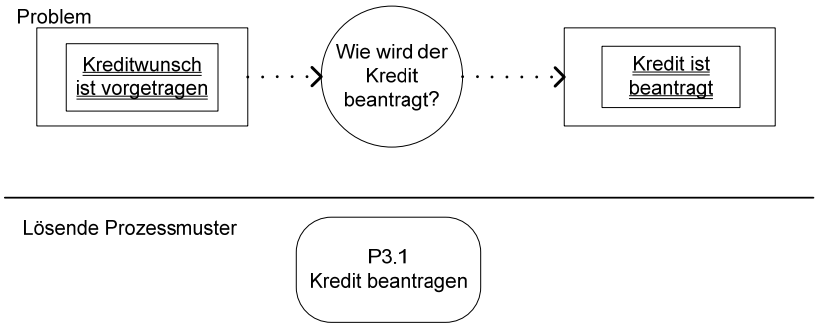



**Abbildung 53: Problemdiagramm "Wie wird der Kredit beantragt?“**

*Der initiale Kontext besteht aus dem Ereignis „Kreditwunsch ist vorgetragen“ (zur linken Seite des Problems) und der resultierende Kontext aus dem Ereignis „Kredit ist beantragt“ (zur rechten Seite des Problems). Das lösende Prozessmuster ist das Prozessmuster P3.1 „Kredit beantragen“.*

*Das Prozessmusterdiagramm zu dem Prozessmuster P3.1 „Kredit beantragen“ ist in Abbildung 54 dargestellt.*

*Dieses Prozessmuster weist in dem Beziehungsdiagramm eine Use-Beziehung zu dem Prozessmustern P3 „Kredithöhe für Hypothekar- oder Privatkredite ermitteln“ auf. Das Prozessmuster P3.1 ist das Komponentenmuster zu P3 (Kompositmuster). Desweiteren besteht eine Sequence-Beziehung zu den Prozessmustern P3.2 „Kredithöhe für Privatkredit ermitteln“ und P3.3 „Kredithöhe für Hypothekarkredit ermitteln“ (Nachfolgermuster). Im Prozessdiagramm ist ein Teilprozess des Prozessmodells model_102 angeführt.*

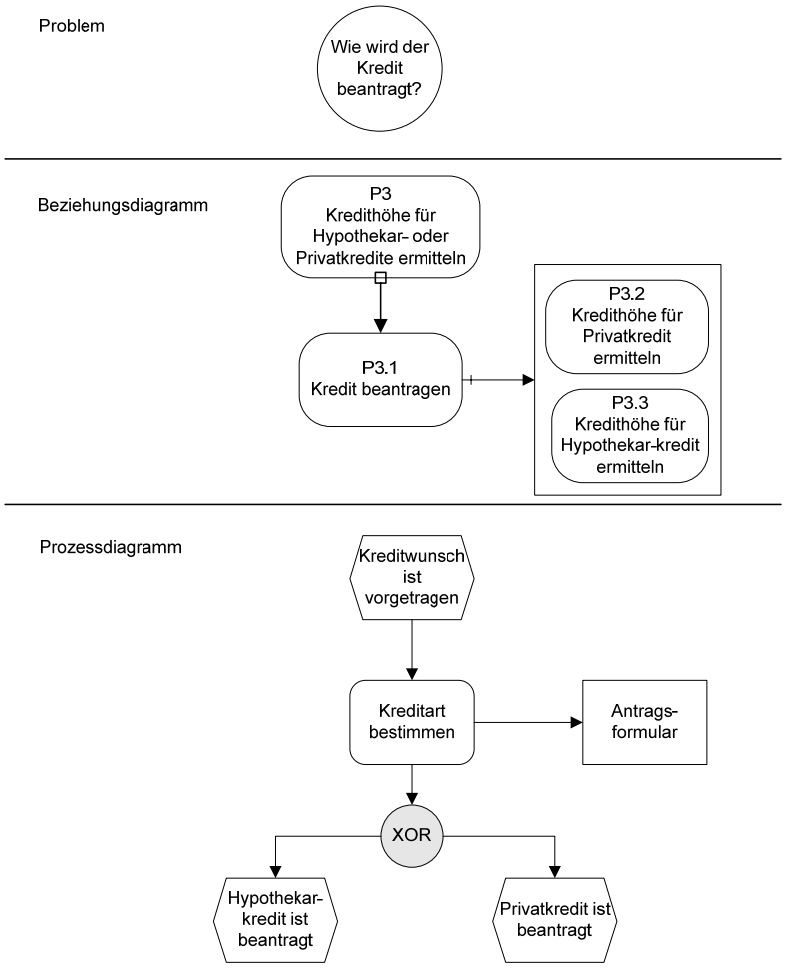



**Abbildung 54: Prozessmusterdiagramm P3.1 "Kredit beantragen"**

*Das Problem „Wie wird die Kredithöhe für den Privatkredit ermittelt?“ wird von dem Prozessmuster P3.2 gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 55 ersichtlich. Der initiale Kontext besteht hier aus dem Startereignis „Privatkredit ist beantragt“ und dem Endergebnis „Kredithöhe ist ermittelt“ als resultierender Kontext.*

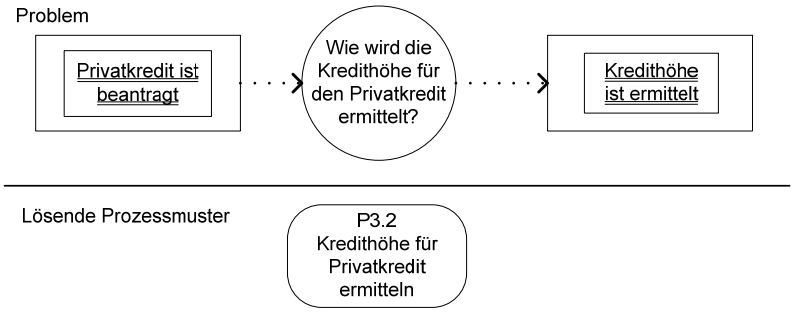



**Abbildung 55: Problemdiagramm "Wie wird die Kredithöhe für den Privatkredit ermittelt?“**

*Abbildung 56 zeigt das Prozessmusterdiagramm zu P3.2 „Kredithöhe für Privatkredit ermitteln“. Das Beziehungsdiagramm weist jeweils eine Use-Beziehung zu den Prozessmustern P3 (Kompositmuster) und P3.2.1 „Kredithöhe für Privatkredit ermitteln (detailliert)“ (Komponentenmuster zu P3.2) und eine Sequence-Beziehung zu P3.1 „Kredit beantragen“ (Vorgängermuster) auf.*

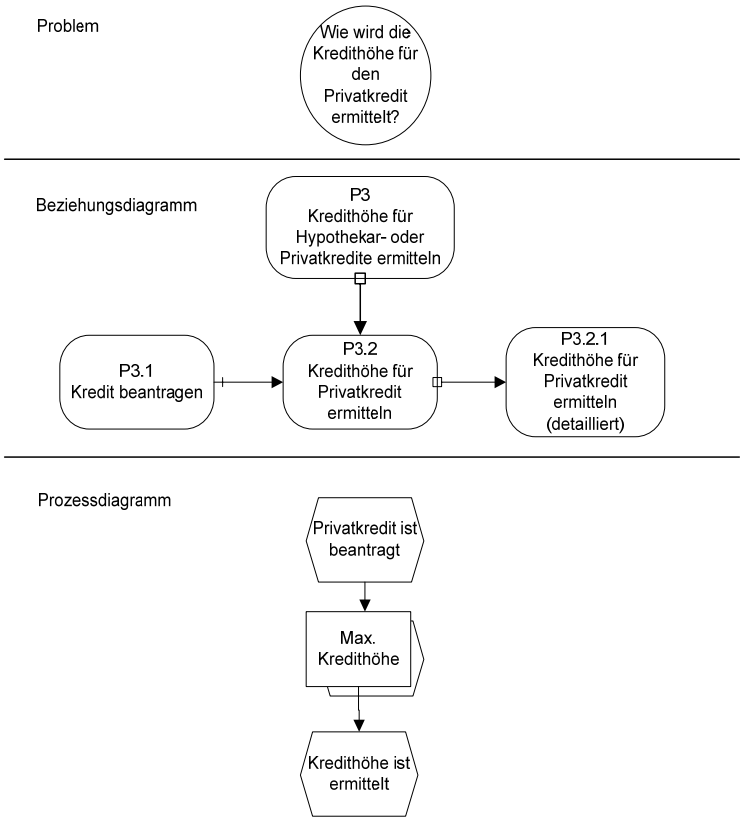



**Abbildung 56: Prozessmusterdiagramm P3.2 "Kredithöhe für Privatkredit ermitteln"**

*Das Problem „Wie wird die Kredithöhe für den Privatkredit ermittelt (detailliert)?“ wird von dem Prozessmuster P3.2.1 gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 57 ersichtlich. Der initiale Kontext besteht hier aus dem Startereignis „Privatkredit ist beantragt“ und dem Endergebnis „Kredithöhe ist ermittelt“ als resultierender Kontext.*

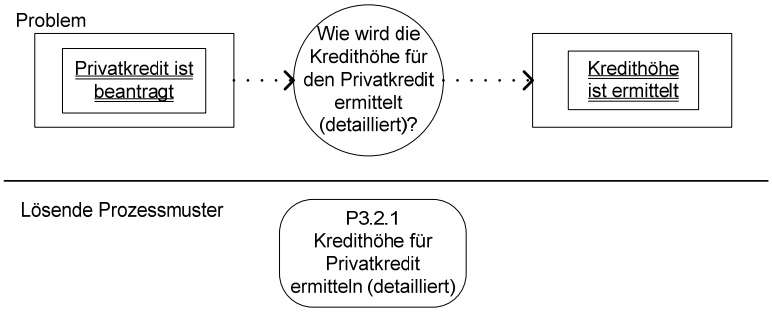



**Abbildung 57: Problemdiagramm "Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“**

*Abbildung 58 zeigt das Prozessmusterdiagramm zu P3.2.1 „Kredithöhe für Privatkredit ermitteln (detailliert)“. Das Beziehungsdiagramm weist jeweils eine Use-Beziehung zu den Prozessmustern P3.2 (Kompositmuster) und zu den Prozessmustern P3.2.1.1 „Unterlagen beschaffen“ und P3.2.1.2 „Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln“ (Komponentenmuster zu P3.2.1) auf. Im Prozessdiagramm wird die ausführlichere Darstellung des Prozesspfades „Max. Kredithöhe“ aus dem Prozessmodell model_1021 dargestellt.*

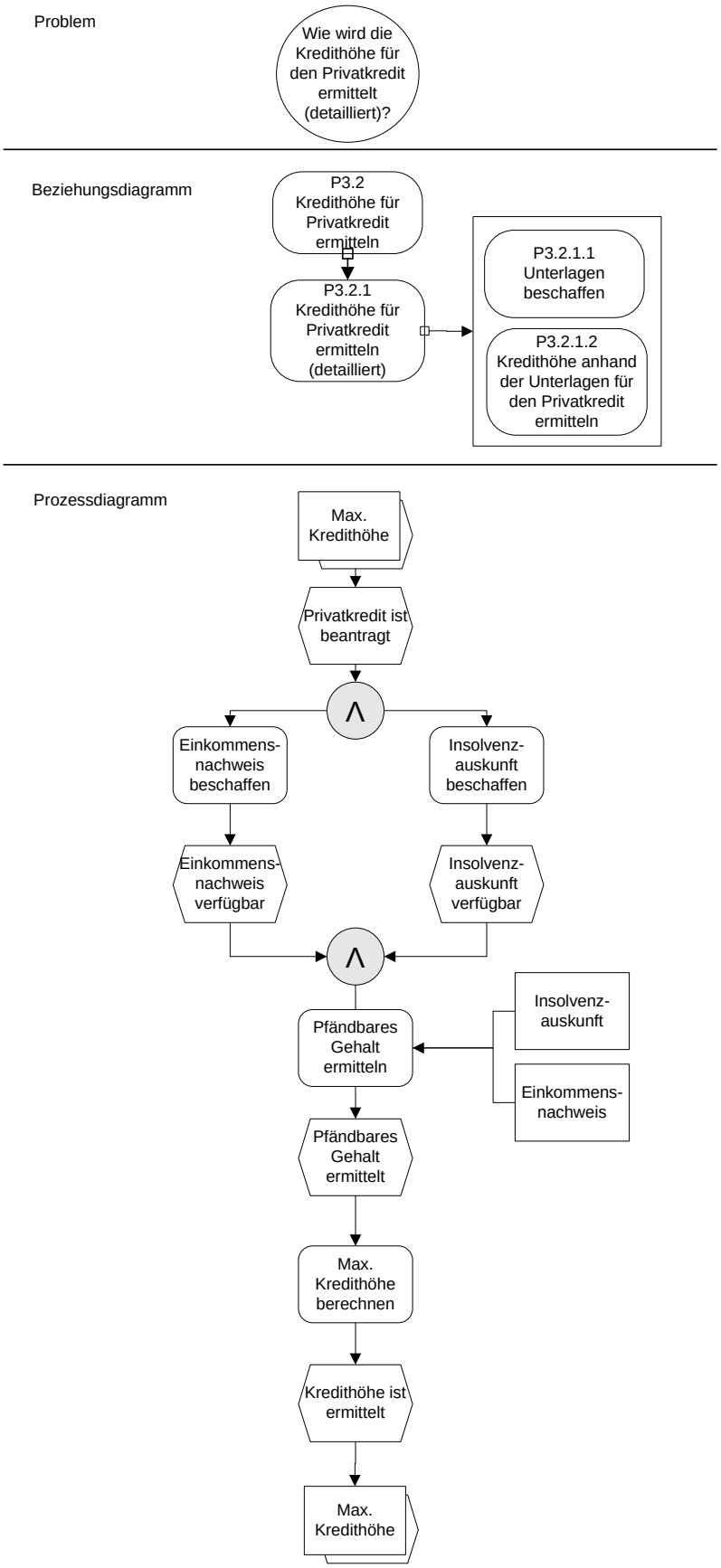



**Abbildung 58: Prozessmusterdiagramm P3.2.1 "Kredithöhe für Privatkredit ermitteln (detailliert)"**

*Das Problem „Wie werden die Unterlagen für den Privatkredit beschafft?“ wird von dem Prozessmuster P3.2.1.1 „Unterlagen beschaffen“ gelöst. Das dazugehörige*

*Problemdiagramm ist in Abbildung 59 ersichtlich. Der initiale Kontext besteht hier ebenfalls aus dem Startereignis „Privatkredit ist beantragt“ und dem resultierenden Kontext „Unterlagen verfügbar“.*

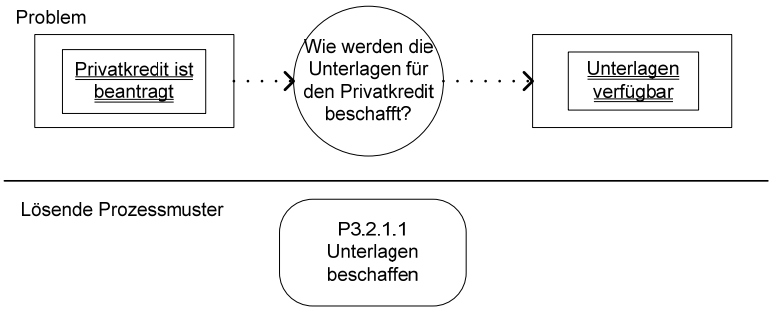



**Abbildung 59: Problemdiagramm "Wie werden die Unterlagen für den Privatkredit beschafft?"**

*Abbildung 60 zeigt das Prozessmusterdiagramm zu P3.2.1.1 „Unterlagen beschaffen“. Das Beziehungsdiagramm weist eine Use-Beziehung zu dem Prozessmuster P3.2.1 als Kompositmuster und eine Sequence-Beziehung zu dem Prozessmuster P3.2.1.2 „Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln“ auf. Im Prozessdiagramm wird ein Teilprozess des Prozessmodells model_1021 dargestellt.*

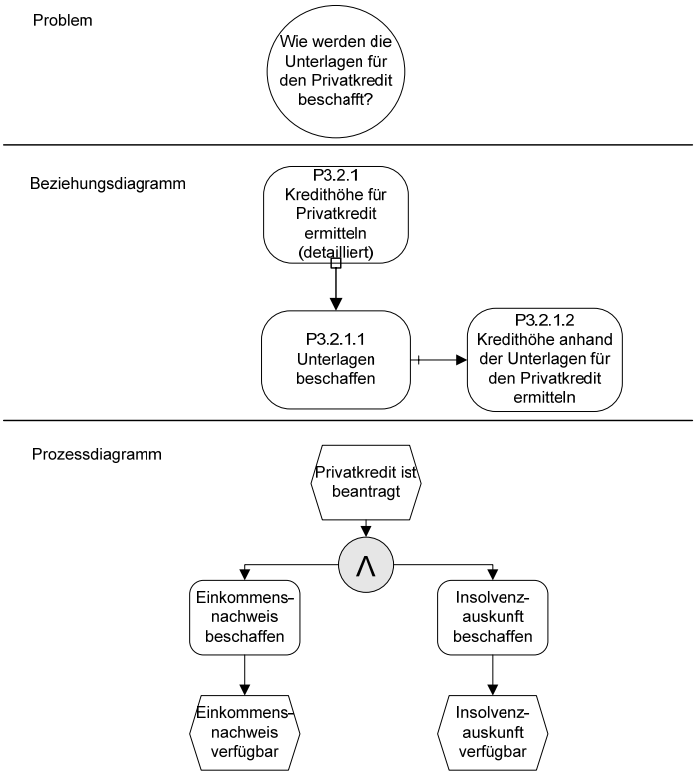



**Abbildung 60: Prozessmusterdiagramm P3.2.1.1 "Unterlagen beschaffen"**

*Das Problem „Wie wird die Kredithöhe für den Privatkredit mit den Unterlagen ermittelt?“ wird von dem Prozessmuster P3.2.1.2 „Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln“ gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 61 ersichtlich. Der initiale Kontext besteht hier aus dem Ereignis „Unterlagen verfügbar“ und dem Endereignis als resultierenden Kontext „Kredithöhe ist ermittelt“.*

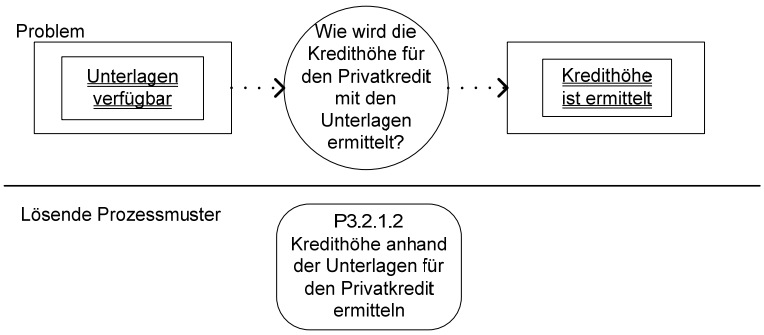



**Abbildung 61: Problemdiagramm "Wie wird die Kredithöhe für den Privatkredit mit den Unterlagen ermittelt? "**

*Abbildung 62 zeigt das Prozessmusterdiagramm zu P3.2.1.2 „Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln“. Das Beziehungsdiagramm weist eine Use-Beziehung zu dem Prozessmuster P3.2.1 als Kompositmuster und eine Sequence-Beziehung zu dem Prozessmuster P3.2.1.1 als Vorgängermuster auf. Im Prozessdiagramm wird ein Teilprozess des Prozessmodells model_1021 dargestellt.*

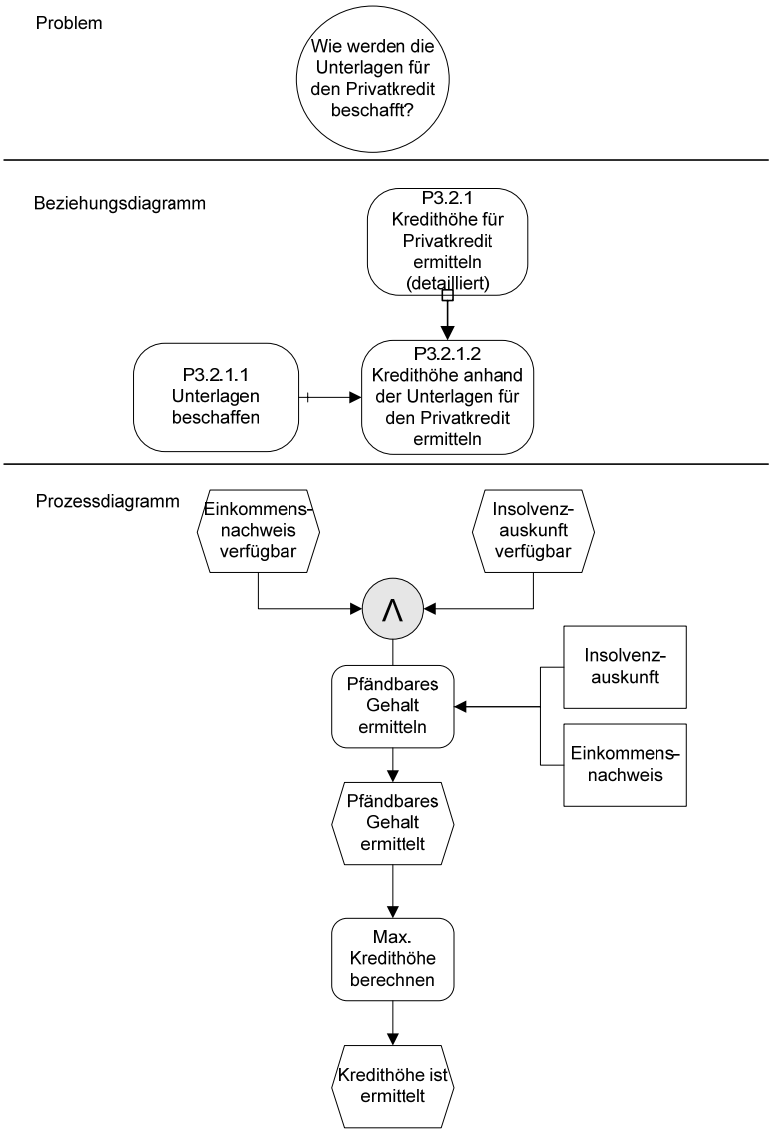



**Abbildung 62: Prozessmusterdiagramm P3.2.1.2 "Kredithöhe anhand der Unterlagen für den Privatkredit ermitteln"**

*Das Problem „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“ wird von dem Prozessmuster P3.3 „Kredithöhe für den Hypothekarkredit ermitteln“ gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 63 ersichtlich. Der initiale Kontext besteht hier aus dem Ereignis „Hypothekarkredit ist beantragt“ und dem Endereignis als resultierenden Kontext „Kredithöhe ist ermittelt“.*

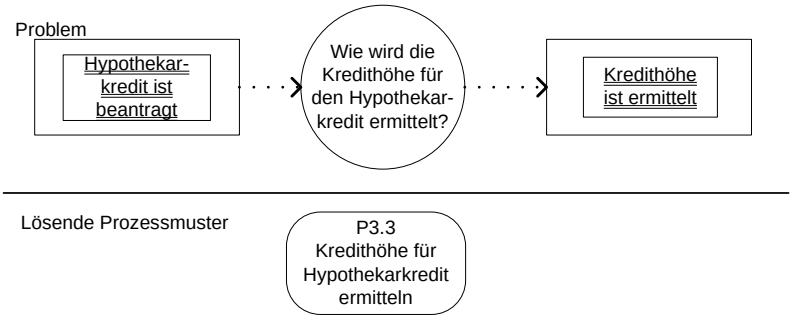



**Abbildung 63: Problemdiagramm "Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“**

*Abbildung 64 zeigt das Prozessmusterdiagramm zu P3.3 „Kredithöhe für den Hypothekarkredit ermitteln“. Das Beziehungsdiagramm weist hier jeweils eine Use-Beziehung zu dem Prozessmuster P3 als Kompositmuster und zu dem Prozessmuster P3.3.1 „Kredithöhe für Hypothekarkredit ermitteln (detailliert) als Komponentenmuster auf. Desweiteren besteht eine Sequence-Beziehung zu dem Prozessmuster P3.1 als Vorgängermuster auf. Im Prozessdiagramm wird ein Teilprozess des Prozessmodells model_102 dargestellt.*

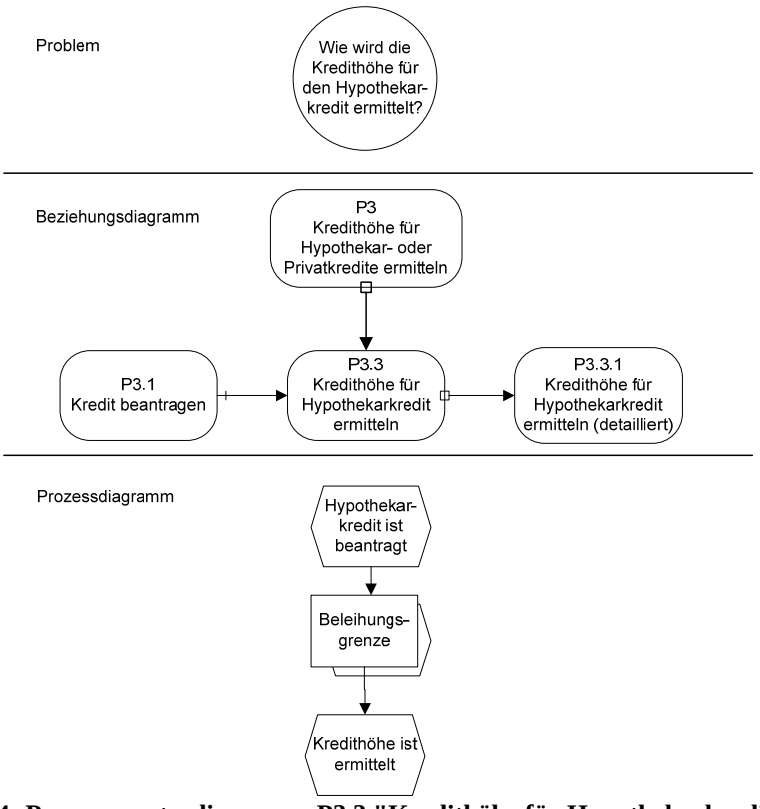



**Abbildung 64: Prozessmusterdiagramm P3.3 "Kredithöhe für Hypothekarkredit ermitteln"**

*Das Problem „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“ wird von dem Prozessmuster P3.3.1 „Kredithöhe für den Hypothekarkredit ermitteln (detailliert)“ gelöst. Das dazugehörige Problemdiagramm*

*ist in Abbildung 65 ersichtlich. Der initiale Kontext besteht hier aus dem Ereignis „Hypothekarkredit ist beantragt“ und dem Endereignis als resultierenden Kontext „Kredithöhe ist ermittelt“.*

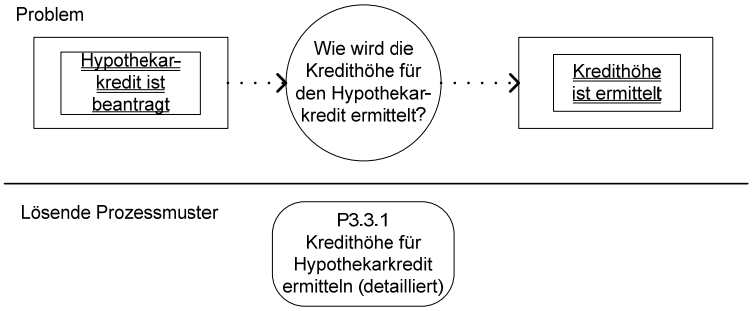



**Abbildung 65: Problemdiagramm "Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?"**

*Abbildung 66 zeigt das Prozessmusterdiagramm zu P3.3.1 „Kredithöhe für Hypothekarkredit ermitteln (detailliert)“. Das Beziehungsdiagramm weist jeweils eine Use-Beziehung zu den Prozessmustern P3.3 (Kompositmuster) und zu den Prozessmustern P3.3.1.1 „Beleihungsgrenze für Hypothekarkredit ermitteln“ und P3.3.1.2 „Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“ (Komponentenmuster zu P3.3.1) auf. Im Prozessdiagramm wird die ausführlichere Darstellung des Prozesspfades „Beleihungsgrenze“ aus dem Prozessmodell model_1022 dargestellt.*

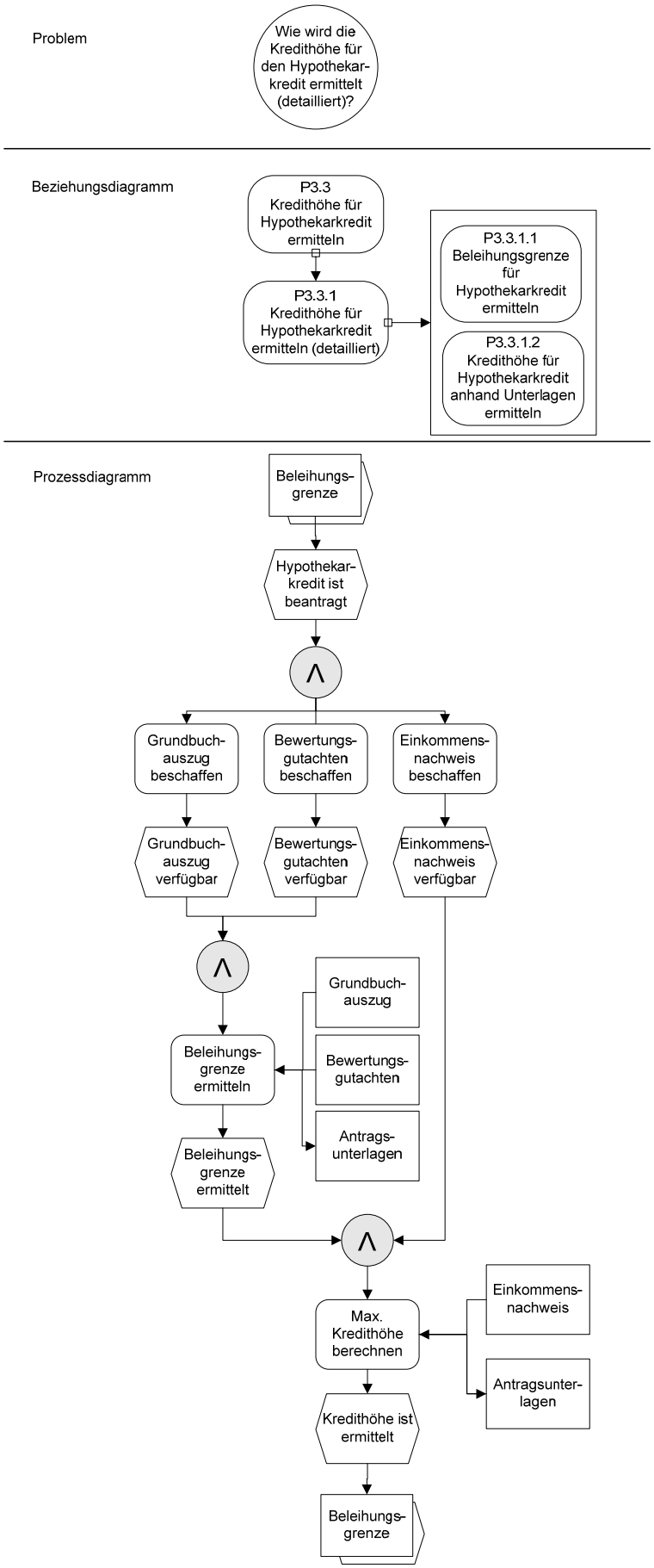



**Abbildung 66: Prozessmusterdiagramm P3.3.1 "Kredithöhe für Hypothekarkredit ermitteln"**

*Das Problem „Wie wird die Beleihungsgrenze für den Hypothekarkredit ermittelt?“ wird von dem Prozessmuster P3.3.1.1 „Beleihungsgrenze für Hypothekarkredit ermitteln“ gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 67 ersichtlich. Der initiale Kontext besteht hier aus dem Ereignis „Hypothekarkredit ist beantragt“ und den Ereignissen „Beleihungsgrenze ermittelt“ und „Einkommensnachweis verfügbar“ als resultierenden Kontext.*

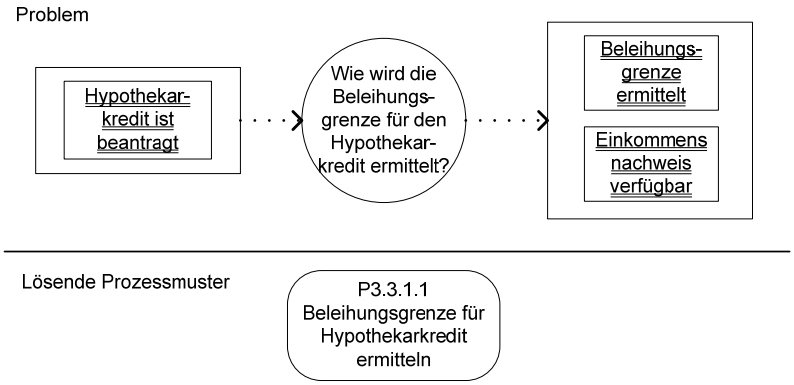



**Abbildung 67: Problemdiagramm "Wie wird die Beleihungsgrenze für den Hypothekarkredit ermittelt?"**

*Abbildung 68 zeigt das Prozessmusterdiagramm zu P3.3.1.1 „Beleihungsgrenze für Hypothekarkredit ermitteln“. Das Beziehungsdiagramm weist eine Use-Beziehung zu dem Prozessmuster P3.3.1 (Kompositmuster) und eine Sequence-Beziehung zu dem Prozessmuster P3.3.1.2 „Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“ als Nachfolgermuster auf. Im Prozessdiagramm wird ein Teilprozess des Prozessmodells model_1022 dargestellt.*

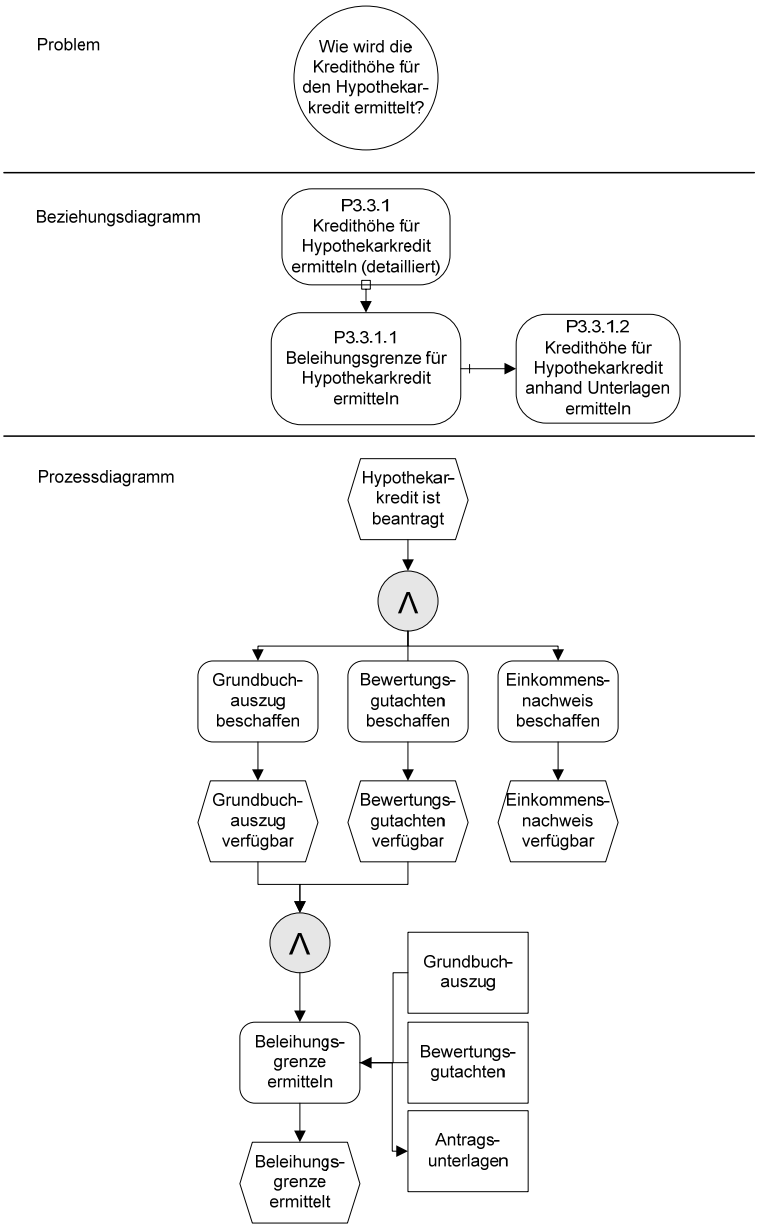



**Abbildung 68: Prozessmusterdiagramm P3.3.1.1 "Beleihungsgrenze für Hypothekarkredit ermitteln"**

*Das Problem „Wie wird die Kredithöhe für den Hypothekarkredit mit Unterlagen ermittelt?“ wird von dem Prozessmuster P3.3.1.2 „Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“ gelöst. Das dazugehörige Problemdiagramm ist in Abbildung 69 ersichtlich. Der initiale Kontext besteht hier aus den Ereignissen „Beleihungsgrenze ermittelt“ und „Einkommensnachweis verfügbar“ und dem Endereignis „Kredithöhe ist ermittelt“ als resultierenden Kontext.*

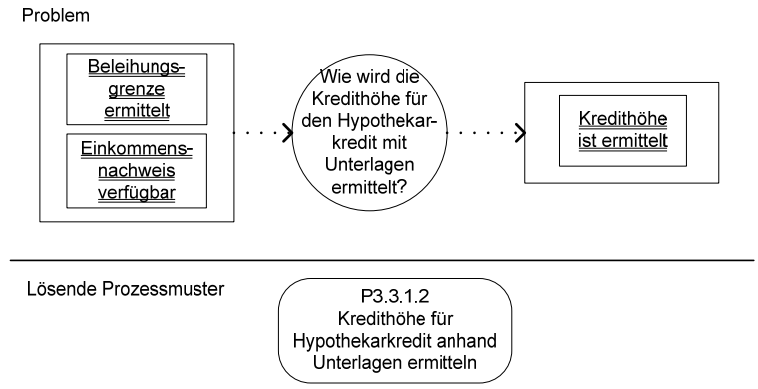



**Abbildung 69: Problemdiagramm "Wie wird die Kredithöhe für den Hypothekarkredit mit Unterlagen ermittelt? "**

*Abbildung 70 zeigt das Prozessmusterdiagramm zu P3.3.1.2 „Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“. Das Beziehungsdiagramm weist eine Use-Beziehung zu dem Prozessmuster P3.3.1 (Kompositmuster) und eine Sequence-Beziehung zu dem Prozessmuster P3.3.1.1 „Beleihungsgrenze für Hypothekarkredit ermitteln“ (Vorgängermuster) auf. Im Prozessdiagramm wird ein Teilprozess des Prozessmodells model_1022 dargestellt.*

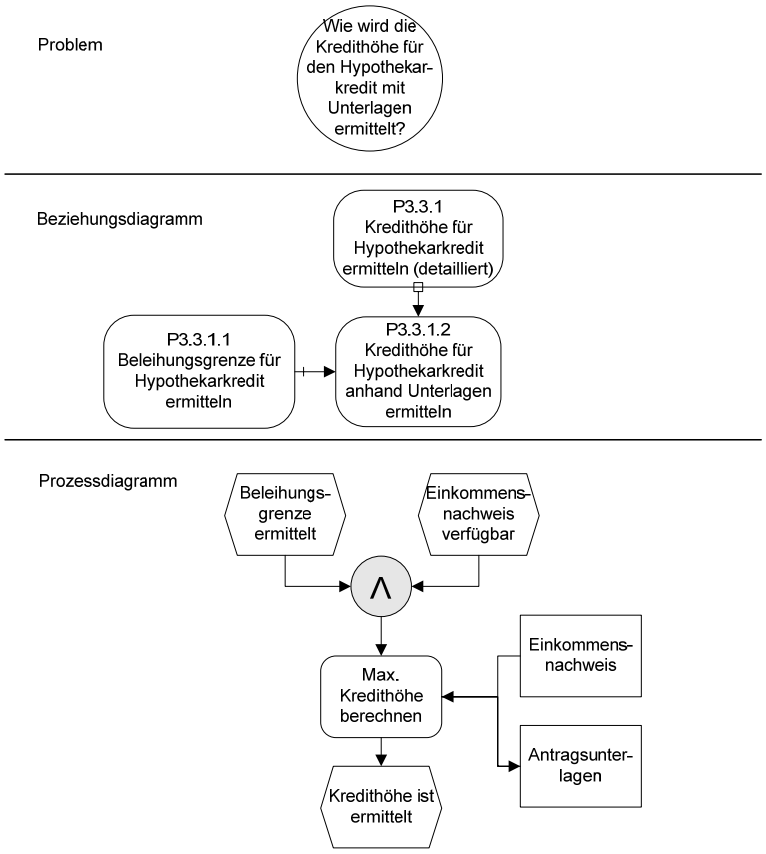



**Abbildung 70: Prozessmusterdiagramm P3.3.1.2 "Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln"**

*Das gesamte Beziehungsdiagramm im Prozessmusterkatalog zu den Prozessmodellen model_102, model_1021 und model_1022 ist in folgender Abbildung zu sehen.*

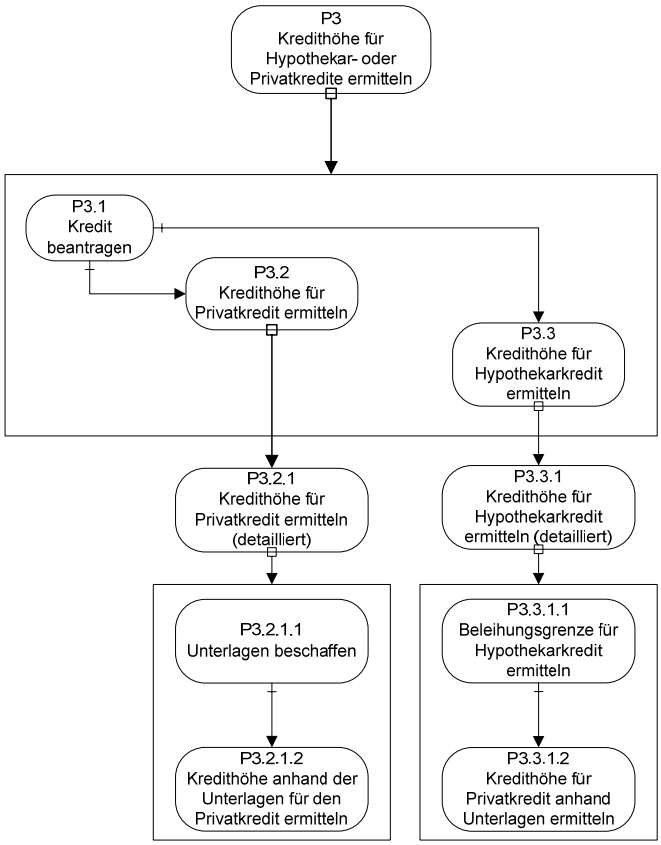



**Abbildung 71: Beziehungsdiagramm model_102, model_1021 und model_1022**

*Folgende neue Pattern Idee kann aus diesen Prozessmodellen erstellt werden: Beispielsweise können für die Prozessmuster P3.2 „Kredithöhe für Privatkredit ermitteln“ und P3.3 „Kredithöhe für Hypothekarkredit ermitteln“ das Supermuster „Kredithöhe ermitteln“ wie in Abbildung 72 im neuen Beziehungsdiagramm dargestellt generiert werden, da von diesen abstrahiert werden kann. Daher kommt das neue Prozessmuster mit einer Refinement-Beziehung hinzu. Man beachte, dass nun die Sequence-Beziehung zwischen dem Prozessmuster P3.1 und dem generalisierten Prozessmuster P3.4 besteht.*

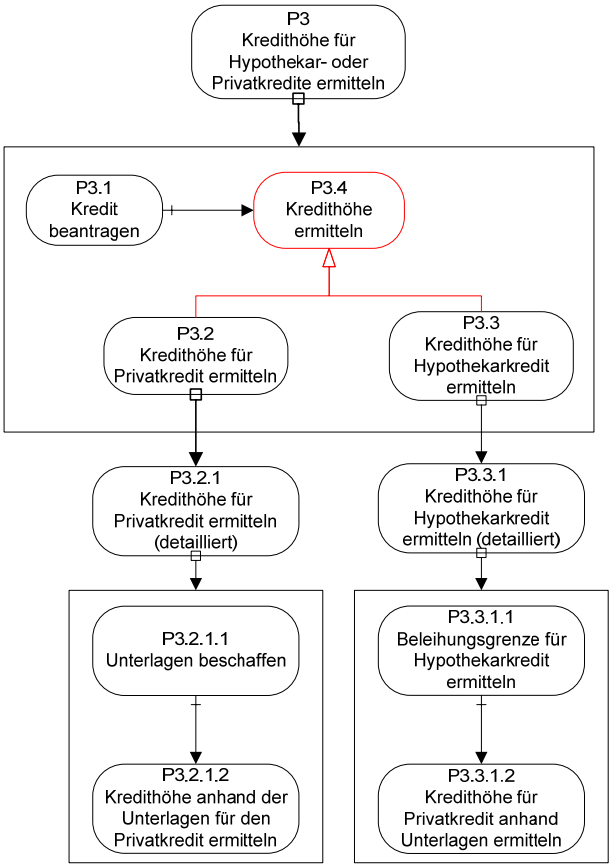



**Abbildung 72: Neues Beziehungsdiagramm model_102, model_1021 und model_1022**

*Das neue Problemdiagramm wird in Abbildung 73 dargestellt. Folgendes Schema wird für das neue Prozessmuster P3.4 „Kredithöhe ermitteln“ identifiziert:*

*P3.4…*

*Name: Kredithöhe ermitteln*
*P… „Wie wird die Kredithöhe ermittelt?“*
*C… Initialer Kontext… „Kredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*
*S… Prozessmodell_3.4*

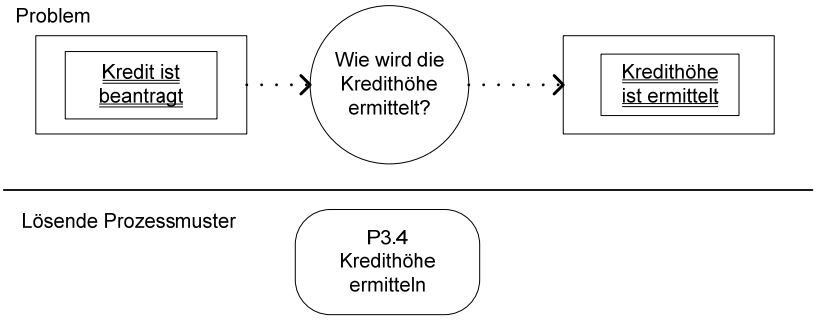



**Abbildung 73: Problemdiagramm "Wie wird die Kredithöhe ermittelt?"**

*Es ergibt sich nun folgendes Prozessmusterdiagramm für das Prozessmuster P3.4 „Kredithöhe ermitteln“:*

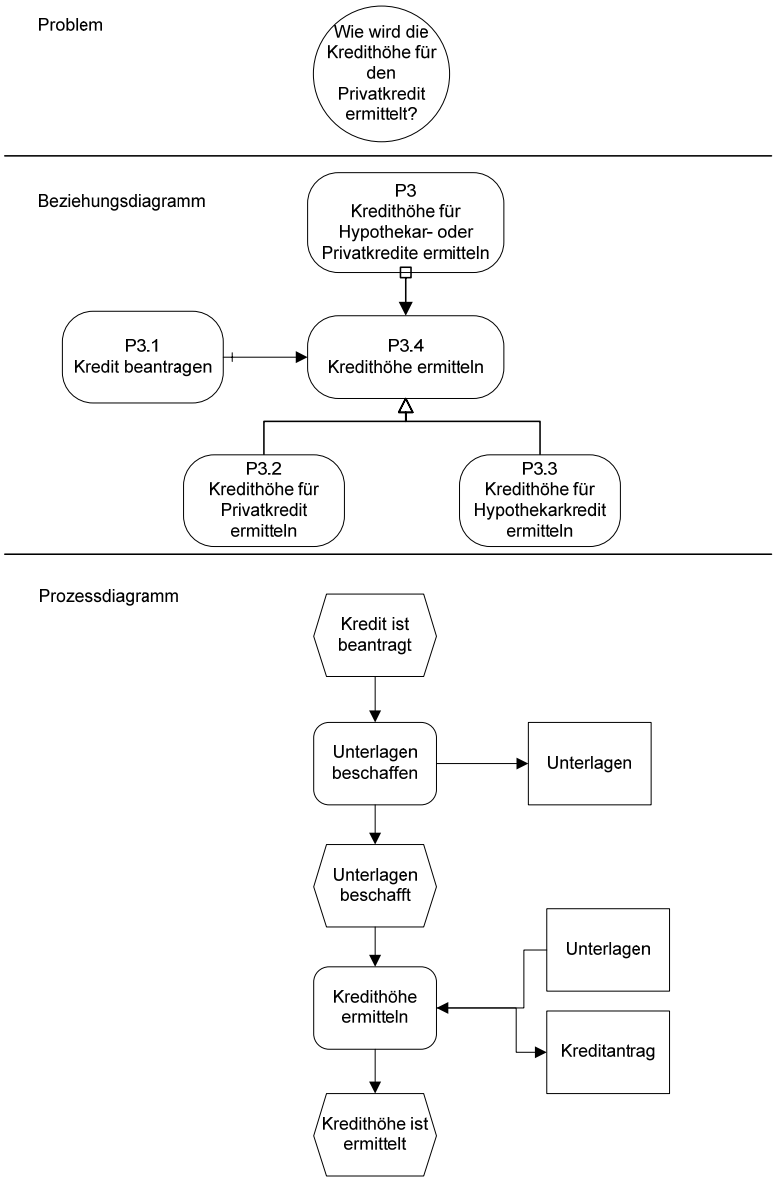



**Abbildung 74: Prozessmusterdiagramm P3.4 „Kredithöhe ermitteln“**

Mit diesem Abschnitt ist die Dokumentation der Prozessmuster abgeschlossen. In den folgenden Abschnitten beginnt der Prozess der reaktiven Wiederverwendung, indem nach geeigneten Prozessmustern gesucht wird, diese angewendet werden und anschließend nach einer Überprüfung eventuell in die Wissensbasis aufgenommen werden.

### 3.4.2 Suche nach geeigneten Prozesslösungen

Die Auswahl von Prozessmustern erfolgt in folgenden Schritten [Hage05, 205f]:

- Problemidentifikation,
- Prozessmustersuche über Prozessmusterbeziehungen und
- Prozessmusterauswahl bei Prozessvarianten.

Der erste Schritt bei diesem Ansatz ist die *„Identifikation eines Problems“*. Dadurch können Prozessmuster im Prozessmusterkatalog gefunden werden, die das identifizierte Problem lösen. Anschließend können über die Prozessmusterbeziehungen weitere Prozessmuster im

Prozessmusterkatalog gesucht werden. Sind mehrere Prozessvarianten zu einem Problem vorhanden, muss der Benutzer das am besten geeignete Prozessmuster anhand selbst gewählter Kriterien auswählen. [Hage05, 205]

***Problemidentifikation:*** Im ersten Schritt identifiziert der Benutzer ein Problem, indem er die Ausgangssituation und die Zielsituation bestimmt. Dazu legt er den initialen und den resultierenden Kontext bestehend aus Objekten und Ereignissen fest. [Hage05, 205]. Folgendes Beispiel soll dies veranschaulichen.

***Beispiel 3.4.2.1***

*Von den unter Abschnitt 3.4.1 erstellten Prozessmustern werden folgende initiale und resultierende Kontexte ausgesucht (in dieser Arbeit nur Ereignisse), um die Aufgabenstellung aus Abschnitt 1.3.1 zu lösen. Dabei werden der initiale und resultierende Kontext fett dargestellt.*

*C…* *Initialer Kontext…* ***„Hypothekarkredit ist beantragt“***
*Resultierender Kontext…* ***„Kredithöhe ist ermittelt“***

*C…* *Initialer Kontext…* ***„Antrag erfasst“***
*Resultierender Kontext…* ***„Kreditantrag bearbeitet“***

*Anschließend können zu diesen Kontexten die dazugehörigen Probleme identifiziert werden.*

*Die identifizierten Probleme zu diesem Kontext sind die Probleme „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“ und „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“.*

*P…* *„Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“*
*C…* *Initialer Kontext…* ***„Hypothekarkredit ist beantragt“***
*Resultierender Kontext…* ***„Kredithöhe ist ermittelt“***

*P…* *„Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“*
*C…* *Initialer Kontext…* ***„Hypothekarkredit ist beantragt“***
*Resultierender Kontext…* ***„Kredithöhe ist ermittelt“***

*Zu dem initialen Kontext „Antrag erfasst“ und „Kreditantrag bearbeitet“ kann folgendes Problem ermittelt werden: „Wie wird der Kreditantrag bearbeitet?“*

*P…* *„Wie wird der Kreditantrag bearbeitet?“*
*C…* *Initialer Kontext…* ***„Antrag erfasst“***
*Resultierender Kontext…* ***„Kreditantrag bearbeitet“***

*Zu den identifizierten Problemen „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“ und „Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“ werden folgende Prozessmuster gefunden:*

*P3.3…*

*Name: Kredithöhe für Hypothekarkredit ermitteln*
*P…* ***„Wie wird die Kredithöhe für den Hypothekarkredit ermittelt?“***
*C…* *Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*

*P3.3.1…*

*Name: Kredithöhe für Hypothekarkredit ermitteln (detailliert)*
*P…* ***„Wie wird die Kredithöhe für den Hypothekarkredit ermittelt (detailliert)?“***
*C…* *Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Kredithöhe ist ermittelt“*

*Zu dem identifizierten Problem „Wie wird der Kreditantrag bearbeitet?“werden mehrere Prozessvarianten gefunden. Die Prozessmuster P0, P1 und P2 lösen auf unterschiedliche Weise das gleiche Problem. Dies kann durch den gleichen initialen und den resultierenden Kontext erkannt werden. Die zu diesen Problemen zugehörigen Prozessmuster sind nachfolgend angeführt.*

*P0…*

*Name: Kreditantrag bearbeiten*
*P…* ***„Wie wird der Kreditantrag bearbeitet?“***
*C…* *Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*

*P1…*

*Name: Kreditantrag bearbeiten*
*P…* ***„Wie wird der Kreditantrag bearbeitet?“***
*C…* *Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*

*P2…*

*Name: Kreditantrag bearbeiten*
*P…* ***„Wie wird der Kreditantrag bearbeitet?“***
*C…* *Initialer Kontext… „Antrag erfasst“*
*Resultierender Kontext… „Kreditantrag bearbeitet“*

***Prozessmustersuche:*** Im zweiten Schritt sind die Prozessmuster selbst das Strukturierungskriterium. Es können weitere Prozessmuster über ihre Beziehungen im Prozessmusterkatalog gesucht werden. [Hage05, 205] Dazu gibt es zwei unterschiedliche Strategien: Top-Down-Strategie und From-Within-Strategie [Hage05, 169].

Wird die *Top-Down-Strategie* verwendet, werden zunächst jene Prozessmuster ausgewählt, die an oberster Ebene der Hierarchie dargestellt sind. Diese Strategie soll bei großen Projekten verwendet werden, da *„eine Festlegung der Prozesse vor Projektbeginn wichtig ist“*. Wie der Name schon sagt, werden nun die in der Hierarchie darunter angesiedelten Prozessmuster ausgewählt, und zwar über die definierten Use-Beziehungen. Im nächsten Schritt können jene in Frage kommenden Prozessmuster ausgewählt werden, die über die Beziehungen „Sequence“, „Refinement“ und „Processvariance“ verbunden sind. [Hage05, 169f]

Mit Hilfe der *From-Within-Strategie* als Bottom-Up-Strategie wird ein in Frage kommendes Prozessmuster ausgewählt, unabhängig in welcher Hierarchiestufe es angesiedelt ist. Anschließend können genutzte Prozessmuster ausgewählt werden, also Prozessmuster die in der Hierarchiestufe weiter unten angesiedelt sind. Diese Strategie ist aufgrund der Flexibilität eher für kleine Projekte geeignet. Sie kann aber auch bei großen Projekten angewendet werden. Es liegt auf der Hand, dass bei großen Projekten zusätzliche Vorgehensmodelle angewendet werden müssen, damit das Projekt nicht aus dem Ruder läuft. Eine sinnvolle Anwendung bei großen Projekten wäre eine Anwendung der From-Within-Strategie bei Änderungen vom Vorgehensmodell. Zu Projektbeginn wird hier nicht wie bei der Top-Down-Strategie eine grobe Richtung vorgegeben, sondern es werden *„je nach Bedarf“* Prozessmuster ausgewählt. [Hage05, 170]

***Beispiel 3.4.2.2***

*Ausgehend von den bereits identifizierten Prozessmustern können nun mit Hilfe der From-Within-Strategie unabhängig von der Hierarchie-Stufe weitere Prozessmuster über die Prozessmusterbeziehungen gesucht werden.*

*Die bereits ausgewählten Prozessmuster sind P3.3 und P3.3.1. Betrachtet man nun die Use-Beziehung des Prozessmusters P3.3.1, so können die genutzten Prozessmuster P3.3.1.1 und P3.3.1.2 ebenfalls ausgewählt werden (vgl. Abbildung 71). Abbildung 75 zeigt die ausgewählten Prozessmuster.*

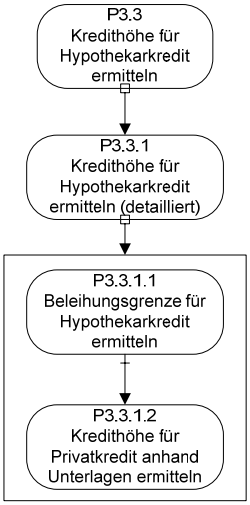



**Abbildung 75: Ausgewählte Prozessmuster zu P3.3.1**

***Prozessmusterauswahl:*** Der Benutzer entscheidet bei mehreren Prozessvarianten selbst, welches Prozessmuster er auswählen will. Dazu kann er einerseits die Prozesslösung betrachten oder weitere Details als Kriterien für die Auswahl hinzuziehen (z.B. „Diskussion“ als Element des Prozessmusterschemas). [Hage05, 205]

***Beispiel 3.4.2.3***

*Zu dem zweiten initialen Kontext „Antrag erfasst“ und dem resultierenden Kontext „Kreditantrag bearbeitet“ werden zu einem Problem drei Prozessmuster als*

*Prozessvarianten gefunden: P0 „Kreditantrag bearbeiten“, P1 „Kreditantrag bearbeiten“ und P2 „Kreditantrag bearbeiten. Angenommen der Benutzer entscheidet sich für das Prozessmuster P0, dann kann er auch hier wieder alle genutzten Prozessmuster aus dem in Abbildung 48 auf Seite 72 dargestellten Beziehungsdiagramm wählen. Folgende Abbildung zeigt die ausgewählten Prozessmuster.*

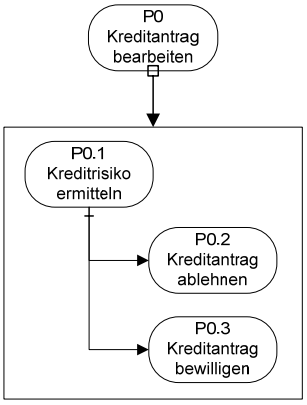



**Abbildung 76: Ausgewählte Prozessmuster zu P0**

### 3.4.3 Anwendung von gefundenen Prozesslösungen

In dieser Phase werden die ausgewählten Prozessmuster angewendet. Sie werden nun vom Benutzer instanziiert, indem zusätzlich benötigte Details wie weitere Prozessschritte, Objekte oder Ereignisse hinzugefügt werden können. Dadurch kann der Benutzer eventuell entdecken, dass die soeben erstellte Prozessmusterinstanz bereits als Prozessverfeinerung des ausgewählten Prozessmusters im Prozessmusterkatalog enthalten ist. [Hage05, 205f] Ist dies nicht der Fall, kann diese Instanz in den Prozessmusterkatalog aufgenommen werden (vgl. Vorgehensweise aus Abschnitt 3.4.1).

***Beispiel 3.4.3.1***

*Die ausgewählten Prozessmuster werden nun instanziiert. Abbildung 77 zeigt das Prozessmuster P3.3 links und die verwendeten Prozessmuster P3.3.1, P3.3.1.1 und P3.3.1.2 rechts der Darstellung. Diese Prozessmodelle können vom Pattern User ohne Veränderung übernommen werden.*

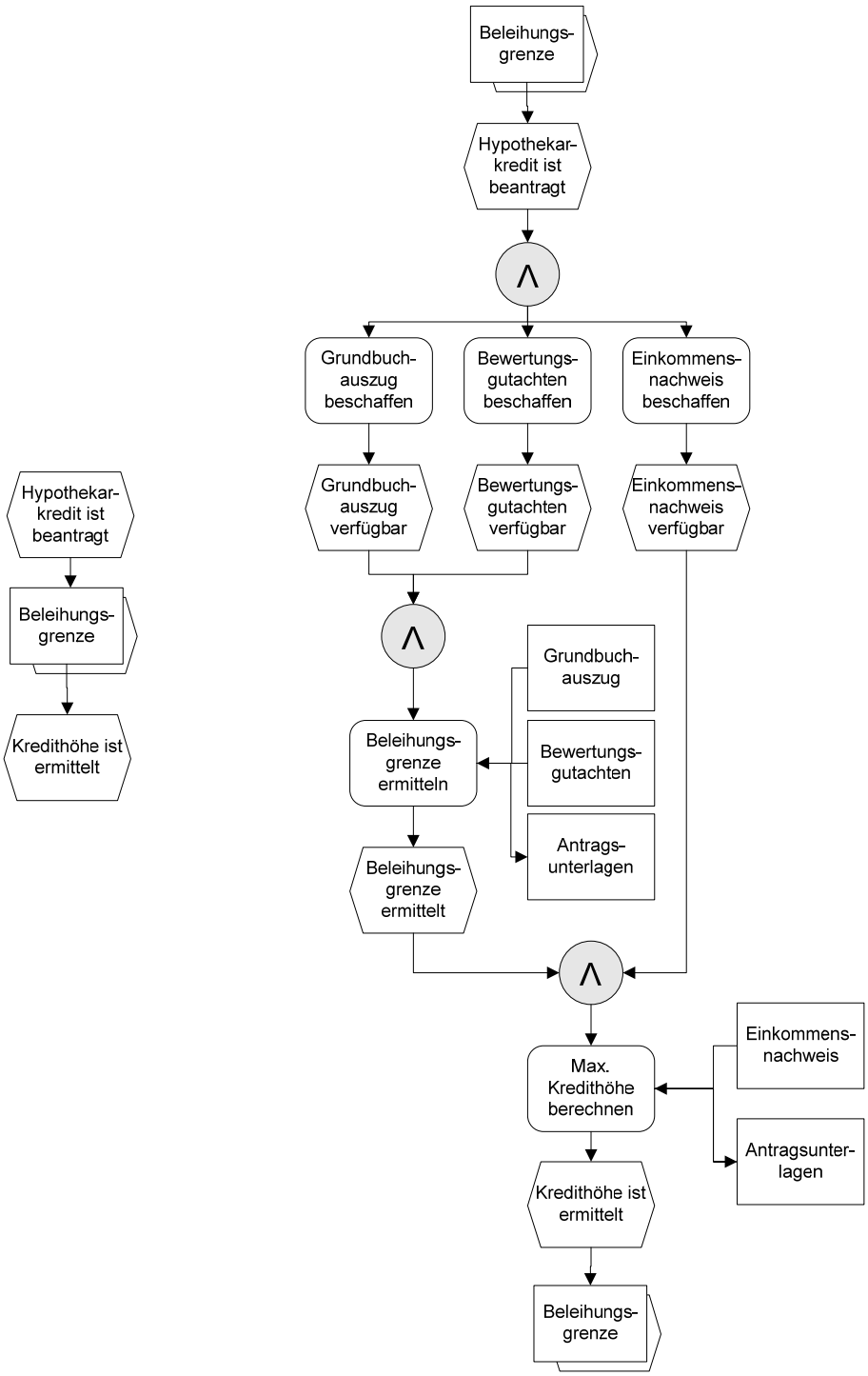



**Abbildung 77: Ausgewählte Prozessmodelle zu P3.3 (links), P3.3.1, P3.3.1.1 und P3.3.1.2 (rechts)**

*Eine Instanziierung kann man in Abbildung 78 entnehmen, wo der abstraktere Begriff „Kredit“ in „Hypothekarkredit“ aus dem Prozessmuster P0 samt den in der Hierarchie weiter unten angeführten Prozessmuster umgeändert werden kann.*

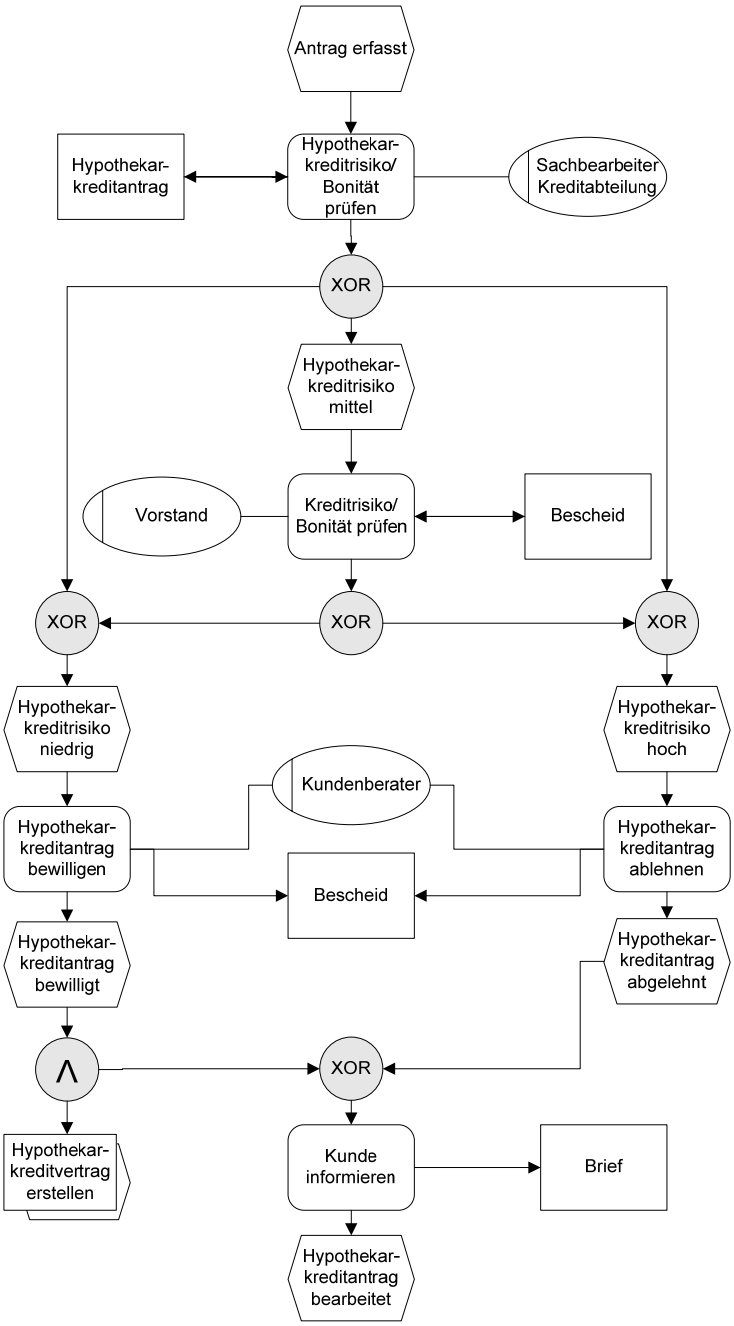



**Abbildung 78: Instanziiertes Prozessmodell zu P0 samt untergeordneten Prozessmustern**

*Abbildung 79 zeigt, wie die beiden Prozessmodelle beispielsweise zusammengeführt werden können, indem ein weiterer Prozessschritt „Antrag ausfüllen“ und ein weiterer Prozesspfad „Hypothekarkredit prüfen“ hinzugefügt wird, um die Übersichtlichkeit zu gewährleisten.*

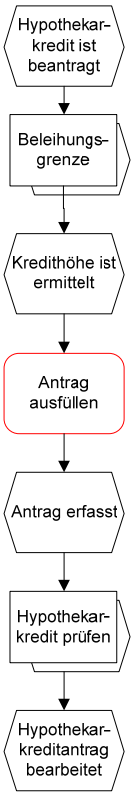



**Abbildung 79: Hinzugefügte Details**

*Dazu muss zu dem instanziierten Prozessmuster P0 das Prozesspfadsymbol ebenfalls angefügt werden (Abbildung 80).*

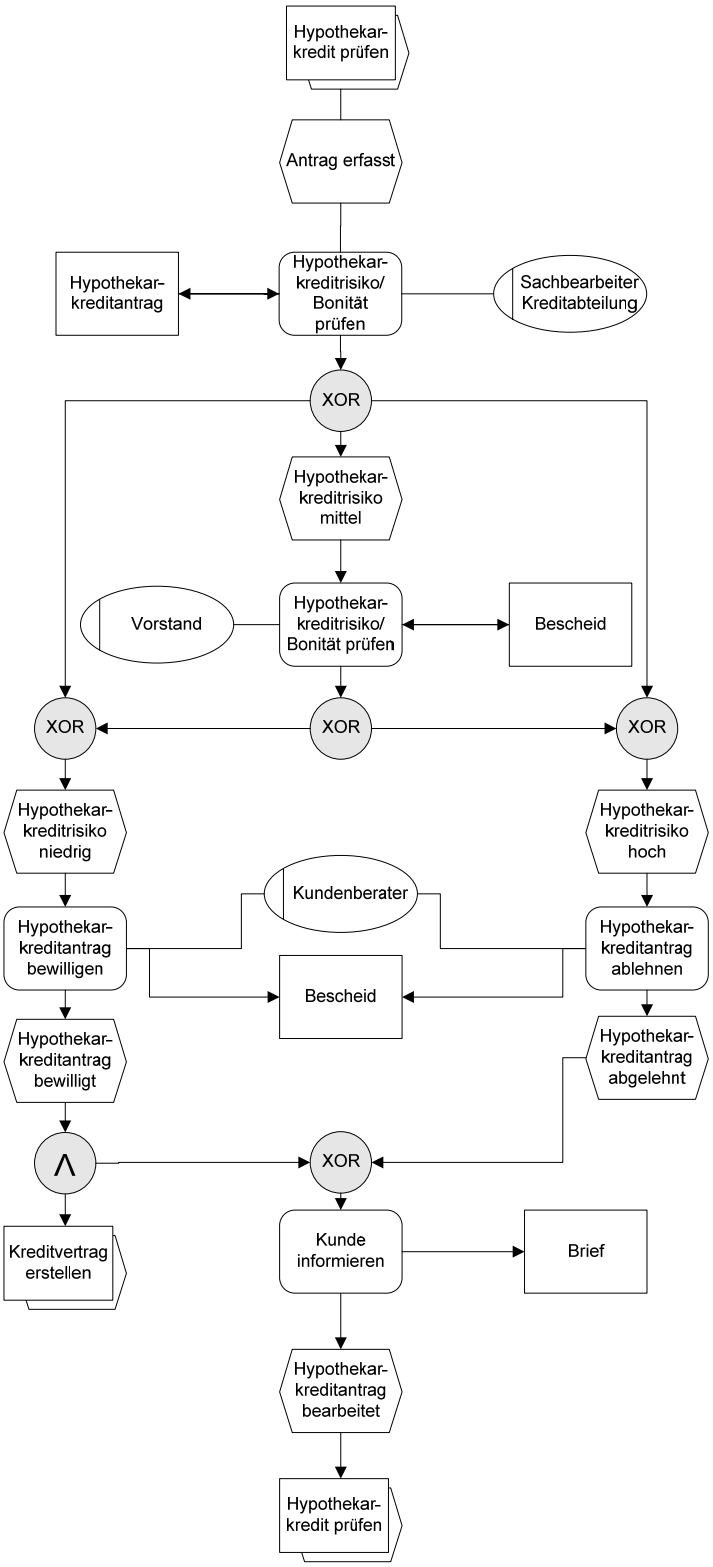



**Abbildung 80: Prozesspfad "Hypothekarkredit prüfen"**

Nach der Anwendung des Prozessmusters kann der Benutzer eine Bewertung für das ausgewählte Prozessmuster abgeben. Dazu können z.B. Kriterien wie Verständlichkeit, korrekte Darstellung in Bezug auf Kontext und Prozess, eventuell fehlende Beziehungen etc. zur Bewertung hinzugezogen werden. Diese Bewertungen sind eventuell Anlass, dass der Pattern Designer entsprechende Änderungen am Prozessmuster vornimmt. [Hage05, 206]

### 3.4.4 Überprüfung der angepassten Prozesslösungen

Die von Hagen eingeführten Konzepte, wie Problem, Prozessmuster, Beziehungen etc. werden auf syntaktische Korrektheit automatisch überprüft [Hage05, 170], wobei die Bedingungen (Kontext) überprüft werden. Werden Prozessmusterbeziehungen festgelegt, wird automatisch eine Überprüfung auf syntaktische Korrektheit angestoßen [Hage05, 178]. Zur Definition der Regeln wurden aus der UML-Spezifikation [UML1.5] die OCL-Regeln (Object Constraint Language) erweitert, um die definierte Syntax einzuschränken [Hage05, 72]. Für eine ausführlichere Definition der Syntax und der OCL-Regeln sei hier auf Hagen verwiesen [Hage05, 69ff]. Im Rahmen dieser Arbeit wurde in den vorangegangenen Abschnitten nur jener Teil der Syntax erklärt, der für die Untersuchung notwendig ist.

### 3.4.5 Integration neuer Prozesslösungen in die Wissensbasis

Wird ein Prozessmuster an den neuen Kontext angepasst, kann das neu entstandene Prozessmuster in die Wissensbasis aufgenommen werden. Dabei müssen die Beziehungen zu den bereits im Prozessmusterkatalog bestehenden Prozessmustern manuell festgelegt werden. [Hage05, 205f] Dies geschieht analog zu der in Abschnitt 3.4.1 dargestellten Dokumentation von Prozessmustern.

***Beispiel 3.4.5.1***

*Zu dem Prozessmodell aus Abbildung 79 können wiederum neue Prozessmuster dokumentiert werden. Dabei wird in folgendes Schema der Name, das „zu lösende Problem“, der initiale und der resultierende Kontext eingetragen.*

*P4…*
*Name: Hypothekarkredit abwickeln*
*P… „Wie wird der vollständige Hypothekarkreditprozess bearbeitet?“*
*C… Initialer Kontext… „Hypothekarkredit ist beantragt“*
*Resultierender Kontext… „Hypothekarkreditantrag bearbeitet“*
*S… Prozessmodell_4*

*Zu diesem Prozessmuster als Kompositmuster wird eine Use-Beziehung zu dem Prozessmuster P3.3 als Komponentenmuster festgelegt, da dieser von ihm genutzt wird (vgl. P3.3 aus Abbildung 77).*

*Da eine zusätzliche Funktion („Antrag ausfüllen“) hinzugefügt wurde, wird zu diesem Teilprozess ebenfalls folgendes Prozessmuster dokumentiert, um eine Beziehung zwischen dem Prozessmuster P3.3 und dem neuen instanziierten Prozessmodell aus Abbildung 80 herstellen zu können (vgl. Abbildung 79).*

*P4.1…*
*Name: Antrag ausfüllen*
*P… „Wie wird der Antrag ausgefüllt?“*
*C… Initialer Kontext… „Kredithöhe ist ermittelt“*
*Resultierender Kontext… „Antrag erfasst“*
*S… Prozessmodell_0.0*

*Das Prozessmuster P4.1 besteht lediglich aus dem Ereignis „Kredithöhe ist ermittelt“ als initialer Kontext, der Funktion „Antrag ausfüllen“ und dem Ereignis „Antrag erfasst“ als resultierender Kontext.*

*Zwischen dem Prozessmuster P4.1 und dem Prozessmuster P4 wird ebenfalls eine Use-Beziehung festgelegt, wobei P4.1 das Submuster ist und P4 das Supermuster darstellt. Zusätzlich kommt zwischen P3.3 und P4.1 eine Sequence-Beziehung hinzu, da P4.1 ein Nachfolgermuster von P3.3 ist (vgl. die Darstellung von P3.3 in Abbildung 77 und P4.1 in Abbildung 79).*

*Zu dem instanziierten Prozessmodell aus Abbildung 80 kann ebenfalls ein neues Prozessmuster dokumentiert werden.*

*P0.0…*
- *Name: Hypothekarkreditantrag bearbeiten*
- *P… „Wie wird der Hypothekarkreditantrag bearbeitet?“*
- *C… Initialer Kontext… „Antrag erfasst“*
  *Resultierender Kontext… „Hypothekarkreditantrag bearbeitet“*
- *S… Prozessmodell_0.0*

*Analog zu der Dokumentation in Abschnitt 3.4.1 kann dieses Prozessmodell wieder in mehrere Teilmodelle aufgeteilt werden und in weitere Prozessmuster zerlegt werden.*

*P0.0.1…*
- *Name: Hypothekarkreditrisiko ermitteln*
- *P… „Wie wird das Hypothekarkreditrisiko ermittelt?“*
- *C… Initialer Kontext… „Antrag erfasst“*
  *Resultierender Kontext… „Hypothekarkreditrisiko ermittelt“*
- *S… Prozessmodell_0.0.1*

*P0.0.2…*
- *Name: Hypothekarkreditantrag ablehnen*
- *P… „Wie wird der Hypothekarkreditantrag abgelehnt?“*
- *C… Initialer Kontext… „Hypothekarkreditrisiko hoch“*
  *Resultierender Kontext… „Hypothekarkreditantrag bearbeitet“*
- *S… Prozessmodell_0.0.2*

*P0.0.3…*
- *Name: Hypothekarkreditantrag bewilligen*
- *P… „Wie wird der Hypothekarkreditantrag bewilligt?“*
- *C… Initialer Kontext… „Hypothekarkreditrisiko niedrig“*
  *Resultierender Kontext… „Hypothekarkreditantrag bearbeitet“*
- *S… Prozessmodell_0.0.3*

*Zwischen dem Prozessmuster P0.0 und P4 wird ebenfalls eine Use-Beziehung festgelegt. Dabei ist wiederum das Prozessmuster P4 das Kompositmuster und P0.0 das Komponentenmuster. Die Prozessmuster P0.0, P0.0.1, P0.0.2 und P0.0.3 stellen eine Verfeinerung der Prozessmuster P0, P0.1, P0.2 und P0.3 dar (vgl. Abbildung 36, Abbildung 39, Abbildung 42 und Abbildung 45 mit Abbildung 80). Daher wird jeweils eine Refinement-Beziehung zwischen P0 und P0.0, zwischen P0.1 und P0.0.1, zwischen P0.2 und P0.0.2 und zwischen P0.3 und P0.0.3 eingefügt. Außerdem wird noch eine*

*Sequence-Beziehung zwischen dem Prozessmuster P4.1 und P0.0 festgelegt, da P0.0 das Nachfolgermuster von P4.1 ist.*

*Auf die Darstellung der neuen Problem- bzw. Prozessmusterdiagramme wird hier verzichtet und der Interpretation des Lesers überlassen. Diese werden nach derselben Vorgehensweise wie in Abschnitt 3.4.1 erstellt.*

# 4 Bausteinbasierter Ansatz nach Rupprecht

Der bausteinbasierte Ansatz nach Rupprecht verwendet *„Prinzipien und Methoden der künstlichen Intelligenz“* und wertet gespeichertes, menschliches Wissen aus, da er insbesondere Erfahrungen der Experten zur Problemlösung hinzuzieht. Daher kann er auch der Kategorie der wissensbasierten Systeme zugeordnet werden und im Spezielleren als Expertensystem bezeichnet werden, einer *„Sonderform der wissensbasierten Systeme“*. [Rupp02, 5]

Ziel dieses Ansatzes ist es, die reaktive Wiederverwendung möglichst flexibel zu unterstützen, da sich die Anforderungen an Projekte bereits während ihrer Laufzeit ändern können. Dieses Konzept beschäftigt sich mit der Unterstützung von projektspezifischen Prozessen, die *„vor allem durch ihre Einmaligkeit und Komplexität gekennzeichnet“* sind. Rupprecht bezeichnet den Vorgang der reaktiven Wiederverwendung in seinem Konzept auch *„Individualisierung von Prozessmodellen“*. Auch projekthafte Prozesse beinhalten Teile und somit Teilprozesse, die sinnvoll wiederverwendet werden können. Diese Teilprozesse werden zu Prozessbausteinen bzw. Musterprozessen zusammengefasst. Ein spezifisches Problem soll somit durch einen oder mehrere Prozessbausteine bzw. Musterprozesse gelöst werden, indem diese in einem bestimmten Zusammenhang gelten (Kontext oder auch Rahmenbedingungen) und durch Regeln (Gestaltungsregeln) in einer zeitlichen und logischen Folge angeordnet werden [Rupp02, 3ff].

Das System generiert Vorschläge zur Anpassung der Prozessmodelle an den jeweiligen Anwendungskontext. Dazu müssen zuerst generische Rahmenbedingungen, generische Prozessmodelle (Prozessbausteine und Musterprozesse) und Gestaltungsregeln definiert werden, mit dessen Hilfe die Vorschläge generiert werden (vgl. Abschnitt 4.3.1).

Der Prozess der reaktiven Wiederverwendung beginnt mit dem Auswählen der Rahmenbedingungen für das aktuelle Projekt. Dabei wird auf *generische Rahmenbedingungen* zurückgegriffen und aus den Einflussgrößen die entsprechenden Ausprägungen ausgewählt. Durch *Gestaltungsregeln* werden die Vorschläge generiert. Dazu werden zu den bereits ausgewählten projektspezifischen Rahmenbedingungen durch *Kontextgestaltungsregeln* weitere Rahmenbedingungen vorgeschlagen (vgl. Abschnitt 4.3.2). Die *Prozessanpassungsregeln* definieren Abhängigkeiten zwischen Rahmenbedingungen und Prozessbausteinen bzw. Musterprozessen. Durch die Prozessanpassungsregeln werden also Prozessbausteine bzw. Musterprozesse abhängig von den ausgewählten Rahmenbedingungen vorgeschlagen. Abhängigkeiten zwischen Prozessbausteinen werden durch *Einbauregeln* definiert, die die Anordnungsbeziehungen zwischen den Prozessbausteinen regeln (vgl. Abschnitt 4.3.3).

Eine Konsistenzprüfung zwischen den gesetzten Rahmenbedingungen und den eingefügten Prozessbausteinen bzw. Musterprozessen kann manuell durch den Benutzer angestoßen werden (vgl. Abschnitt 4.3.4).

Die Wissensbasis kann laufend von den Anwendern mit der Wissenserwerbskomponente verwaltet und erweitert werden (vgl. Abschnitt 4.3.5).

Bevor die Vorgehensweise der Wiederverwendung detailliert anhand des Rahmenbeispiels erklärt wird, werden zuerst die verwendeten Konzepte erläutert.

## *4.1 Verwendete Konzepte*

Prozessbausteine sind *„kleinere, zeitlich und logisch in sich abgeschlossene Einheiten“* und werden aus Prozessmodellen herausgelöst oder ohne Verwendung von Prozessmodellen neu gestaltet. Prozessbausteine können in Teilprozesse über *„beliebig viele Ebenen“* aufgeteilt werden (Dekomposition), jedoch *„keine anderen Prozessbausteine enthalten“*. Für diese Teilprozesse können anschließend ebenfalls neue Prozessbausteine definiert werden, die *„unter einem neuen Namen“* abgelegt werden müssen. Für diesen Fall gibt es keine Beziehung zwischen dem neu erstellten Prozessbaustein und dem als Vorlage dienenden Prozessbaustein. [Rupp02, 83ff]

**Musterprozesse**

Musterprozesse dienen als Vorschläge für Teilprojekte und sind logisch zusammengehörige Arbeitspakete. Sie können Prozessbausteine und/oder andere Musterprozesse enthalten. Daher sind sie in der Regel größer als Prozessbausteine. Sie bilden *„ein Grundgerüst an Aktivitäten und Teilprozessen“*. Die Prozessbausteine und/oder Musterprozesse können aus dem Musterprozess entfernt werden. Somit machen Prozessbausteine und oder die Musterprozesse *„den variablen Anteil“* des Musterprozesses aus. [Rupp02, 84]

**Rahmenbedingungen**

Um die Prozessmodelle zu individualisieren, das heißt die Prozessmodelle an einen neuen Kontext anzupassen, wird das Konzept der *Rahmenbedingungen* eingeführt. Die Rahmenbedingungen werden durch Einflussgrößen und ihren Ausprägungen repräsentiert und stellen somit den Kontext dar, in dem das Projekt zu sehen ist. [Rupp02, 5] Rupprecht unterscheidet dabei zwischen generischen und projektspezifischen Rahmenbedingungen. Die *generischen Rahmenbedingungen* dienen als Basis für die *projektspezifischen Rahmenbedingungen*. [Rupp02, 67ff]

**Gestaltungsregeln**

Durch Gestaltungsregeln wird die Lösung schlussendlich als Vorschlag präsentiert. Es wird zwischen Kontextgestaltungsregeln, Prozessanpassungsregeln und Einbauregeln unterschieden. Durch *Kontextgestaltungsregeln* können weitere Rahmenbedingungen hinzugefügt werden, die der Benutzer nicht ausgewählt hat, jedoch Abhängigkeit zwischen diesen repräsentieren. *Prozessanpassungsregeln* definieren Abhängigkeiten zwischen den Rahmenbedingungen sowie den verknüpften Prozessbausteinen bzw. Musterprozessen. Dadurch können Prozessbausteine hinzugefügt oder aus Musterprozessen entfernt werden. Bei den Kontextgestaltungsregeln und Prozessanpassungsregeln können mehrere Ausprägungen der Rahmenbedingungen als Prämissen durch eine UND-Verknüpfung verbunden werden. Die *Einbauregeln* bestimmen die *„Anordnungsbeziehungen zwischen Prozessbausteinen“*. Dabei sind *„ist Vorgänger von“*, *„ist Nachfolger von“*, *„ist parallel zu“* und *„ist zugehörig zu“* die gültigen Beziehungstypen (Aktionstypen). Einbauregeln verknüpfen immer zwei Prozessmuster miteinander. [Rupp02, 73, 89]

Folgende Abbildung stellt die möglichen Aktionstypen dieser Gestaltungsregeln zwischen den Rahmenbedingungen, zwischen Rahmenbedingung und Prozessbaustein und zwischen den Prozessbausteinen grafisch dar.

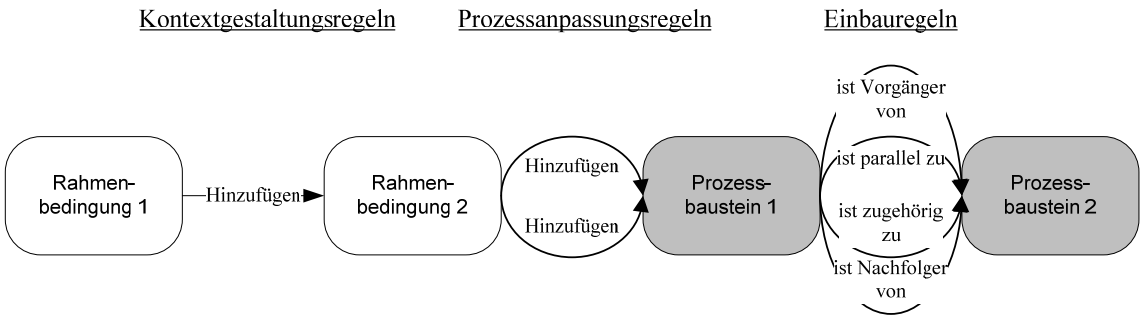



**Abbildung 81: Gestaltungsregeln mit Aktionstypen [Rupp02, 90]**

## Lösung

Die Lösung wird anhand der Rahmbedingungen in Verbindung mit Gestaltungsregeln aus generischen Prozessmodellen zusammengesetzt, die entweder Musterprozesse oder Prozessbausteine sind und einzelne Prozessschritte beinhalten [Rupp02, 75].

Der in der Abbildung 82 dargestellte Prozessbaukasten veranschaulicht das Zusammenspiel der verwendeten Konzepte. Die *projektspezifischen Prozessmodellinstanzen* sind instanziierte Prozessmodelle. Zusammen mit den *Projektrahmenbedingungen* (projektspezifischen Rahmenbedingungen), die den Kontext spezifizieren, in dem sie gelten, ergeben sie einen *Prozessfall*, der somit *„ein konkretes Projekt“* darstellt. Aus den projektspezifischen Prozessmodellinstanzen werden Prozessbausteine und eventuell auch Musterprozesse definiert sowie die *generischen Rahmenbedingungen* um die nicht vorhandenen projektspezifischen Rahmenbedingungen erweitert. Im Zuge dieser Tätigkeiten werden die geltenden *Gestaltungsregeln* definiert (Kontextanpassungsregeln, Prozessanpassungsregeln und Einbauregeln). Für die Erstellung eines neuen Prozessfalles, und damit die Durchführung des Prozesses der reaktiven Wiederverwendung, werden aus den *generischen Rahmenbedingungen* projektspezifische Rahmenbedingungen gewählt. Durch die definierten Gestaltungsregeln können eventuell weitere Rahmenbedingungen hinzugefügt werden (Kontextgestaltungsregeln), anschließend die betroffenen Prozessbausteine dadurch identifiziert werden (Prozessanpassungsregeln) und eventuell weitere Prozessbausteine hinzugefügt werden (Einbauregeln). [Rupp02, 72f]

In Anlehnung an Expertensysteme werden diese Aufgaben der Wissenserwerbskomponente, Faktenbeschreibungskomponente und Problemlösungs- und Erklärungskomponente zugewiesen.

Die *Wissenserwerbskomponente* liefert Funktionen, um neue Prozessbausteine, Rahmenbedingungen und Gestaltungsregeln zu definieren sowie sie zu pflegen und zu warten. [Rupp02, 5] Mit dieser Komponente werden demnach unter anderem der Wissenserwerb und das Speichern des neu geschaffenen Wissens durchgeführt (*Integration von externen Prozesslösungen* und *Integration von neuen Prozesslösungen in die Wissensbasis*).

Die *Faktenbeschreibungskomponente* dient dazu, Projektrahmenbedingungen zu setzen, wodurch das neu zu erstellende Projekt auf den projektspezifischen Kontext ausgerichtet wird. Dabei sind die generischen Rahmenbedingungen die Grundlage, die in der Wissenserwerbskomponente definiert wurden. [Rupp02, 5] Dieser Komponente kann ein Teil der Suche nach Prozesswissen zugeordnet werden, da durch das Festlegen der Rahmenbedingungen die Suche angestoßen wird (Vorbereitung zur *Suche nach geeigneten Prozesslösungen*).

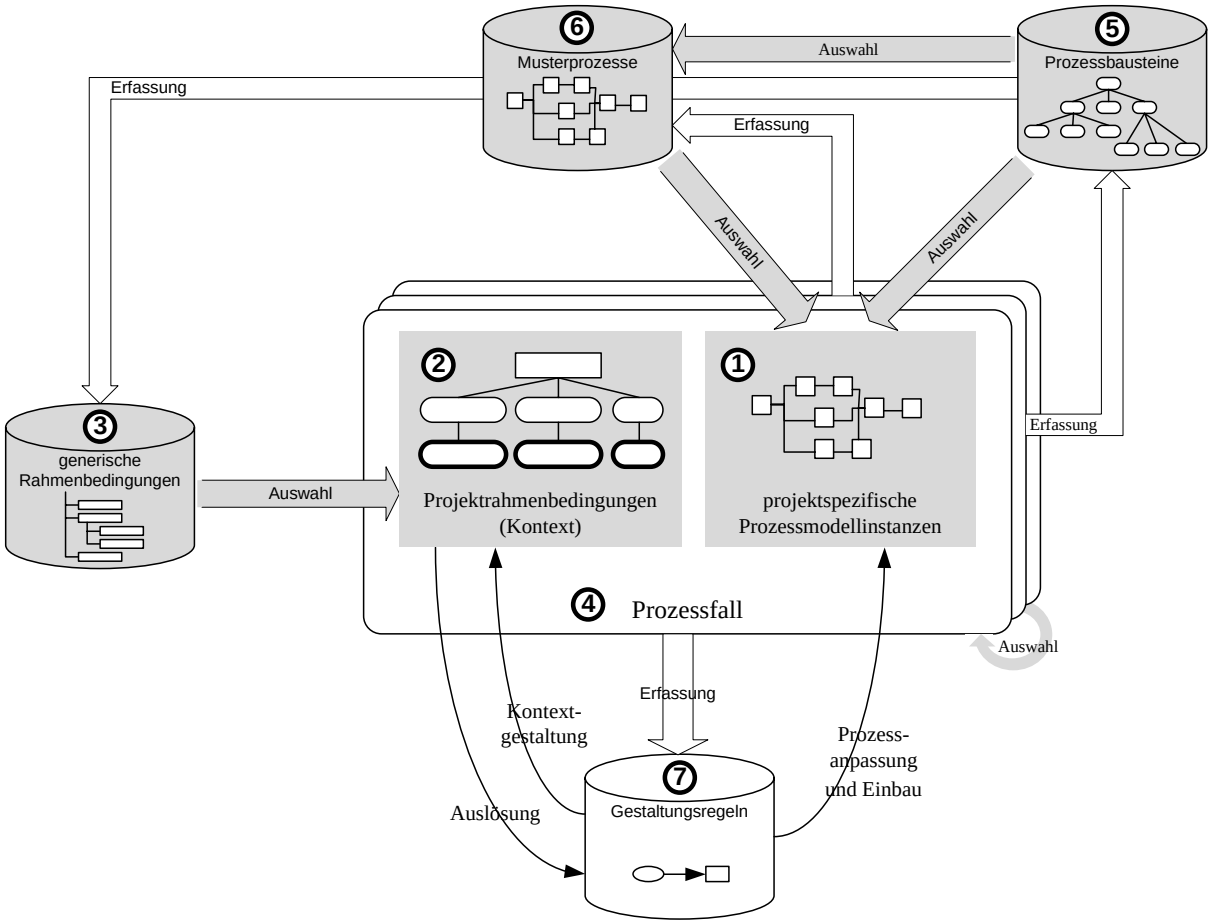



**Abbildung 82: Prozessbaukasten [RRFS01]**

Die *Problemlösungs- und Erklärungskomponente* generiert Vorschläge zur *„Individualisierung der projektspezifischen Prozessmodelle“*. Mit dieser Komponente wird nach der Lösung gesucht und auch eine Konsistenzprüfung zwischen den gesetzten Rahmenbedingungen und den eingefügten Prozessmodellinstanzen durchgeführt, wenn diese vom Benutzer angestoßen wird. Demnach werden die gesetzten projektspezifischen Rahmenbedingungen ausgewertet und die entsprechenden Prozessbausteine oder Musterprozesse eingefügt. [Rupp02, 5, 117ff] Dieser Komponente können demnach folgende Aufgaben zugeordnet werden: *Suche nach geeigneten Prozesslösungen*, *Anwendung von gefundenen Prozesslösungen* und *Überprüfung der angepassten Prozesslösungen*.

Abbildung 83 gibt einen Überblick über von Rupprecht ausgewählte Funktionen, von denen die für die Untersuchung notwendigen beschrieben werden. Diese werden den jeweiligen Komponenten (Wissenserwerbskomponente, Faktenbeschreibungskomponente und Problemlösungs- und Erklärungskomponente) zugewiesen [Rupp02, 93].

Auf die Funktionen der Wissenserwerbskomponente, der Faktenbeschreibungskomponente und der Problemlösungs- und Erklärungskomponente wird in Abschnitt 4.3 näher eingegangen, in dem das Vorgehensmodell unter Verwendung dieser Funktionen erklärt wird.

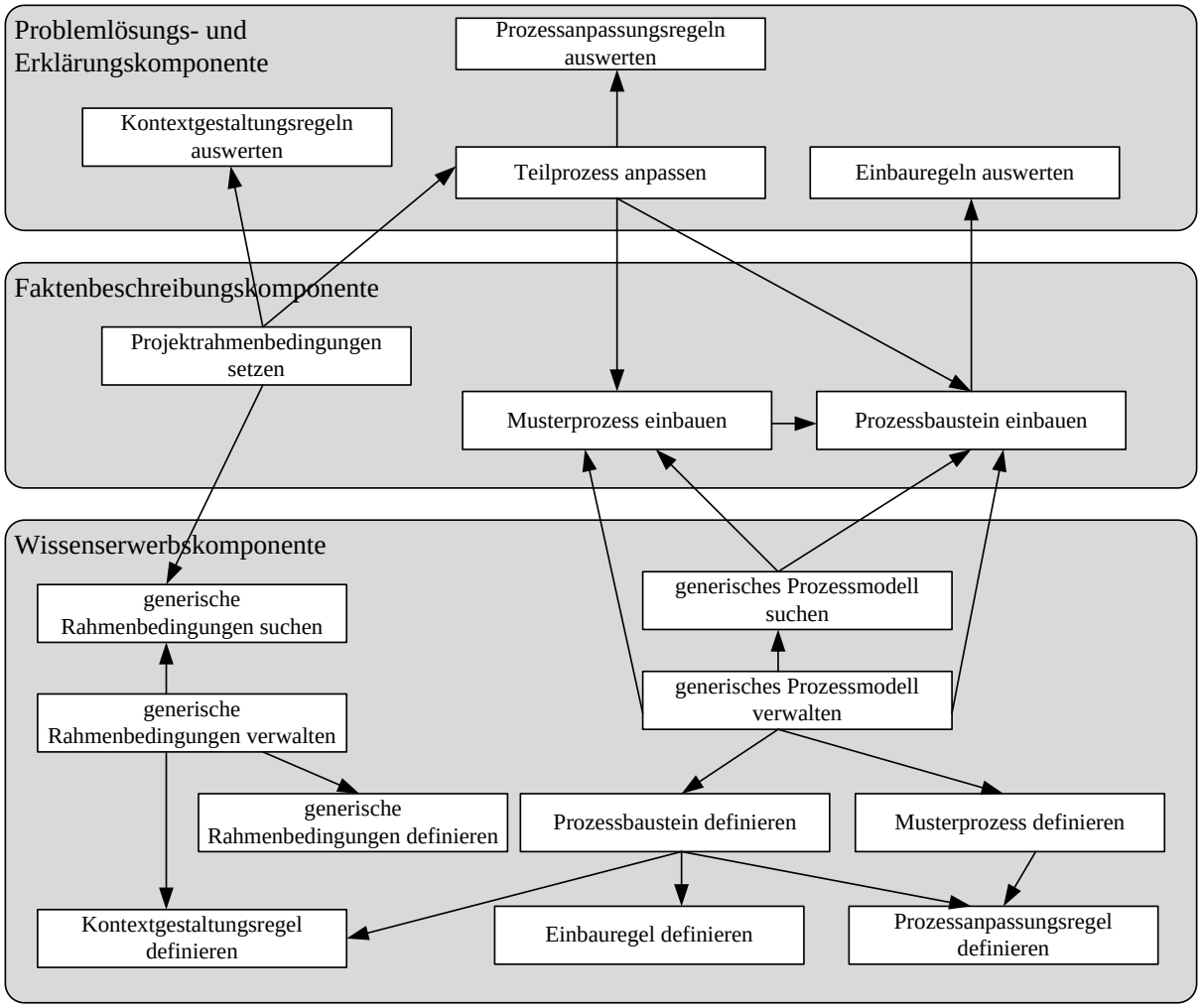



**Abbildung 83: Auszug aus den Funktionen der einzelnen Komponenten**[2]

## *4.2 Organisation der Wissensbasis*

Grundlage für diesen Ansatz ist eine strukturierte Organisation der Wissensbasis, die mit Hilfe der Funktionen der Wissenserwerbskomponente durchgeführt wird, um implizite Erfahrungen der Projektbeteiligten explizit zu erfassen. Die Wissensbasis setzt sich bei diesem Ansatz aus jeweils einer Bibliothek für generische Prozessmodelle (Prozessbausteine und Musterprozesse mit Prozessbausteinen), und generische Rahmenbedingungen zusammen. [Rupp02, 38ff, 85ff]

**Bibliothek für generische Prozessmodelle**

Die Anordnung der Prozessbausteine und Musterprozesse erfolgt innerhalb einer mehrstufigen Ordnerhierarchie [Rupp02, 85]. Zur Typologisierung der Ordnerstruktur existieren verschiedene methodische Vorgehensweisen bzw. Kriterien [Lang97, 91]. In diesem bausteinbasierten Ansatz wird ein Gliederungsschema auf oberster Ebene vorgegeben (deduktiv), *„auf den unteren Ebenen“* der Hierarchie können die Anwender die Kriterien aufgrund ihrer Beobachtungen selbst hinzufügen [Rupp02, 85].

[2] Mit Änderungen in der grafischen Darstellung übernommen aus [Rupp02, 93]

Zur Typologisierung stehen folgende Konzepte zur Verfügung [Rupp02, 85]:

- Prozesskategorien: Die Prozessbausteine werden Prozesskategorien zugeordnet. Diese können einerseits Prozessmodelle, andererseits weitere Unterkategorien enthalten. Dabei kann *„eine Kategorie auch in mehreren Oberkategorien enthalten sein“*. Dies trifft auch auf die Musterprozesse und Prozessbausteine zu, die ebenfalls *„in mehreren Prozesskategorien enthalten sein können“*.

- Generalisierungs-/Spezialisierungsbeziehung zwischen Prozessmodellen: Zwischen den Prozessmodellen können Generalisierungs- bzw. Spezialisierungsbeziehungen bestehen. Dies bedeutet, dass Prozessvarianten, also alternative Prozesslösungen als Prozessbausteine bzw. Musterprozesse, verknüpft werden können, die jedoch *„nicht konsistent gehalten“* werden. Änderungen eines Prozessbausteins führen somit nicht zu Änderungen des abstrakten Prozessbausteins und die *„Strukturen und Eigenschaften“* der abstrakten Prozessbausteine werden nicht an den spezielleren Prozessbaustein vererbt.

Abbildung 84 zeigt ein Beispiel einer Ordnerstruktur für Prozessbausteine. Die Rechtecke entsprechen den Prozesskategorien, wobei Ober- und Unterkategorien angeführt sind. Die Beziehungen zwischen Ober- und Unterkategorien werden mit einer dickeren Linie als die Generalisierungs-/Spezialisierungsbeziehungen zwischen den Prozessbausteinen dargestellt. Die speziellen Prozessbausteine stellen Varianten dar. [Rupp02, 86f] Der Prozesskategorie „Abwicklung“, die der Prozesskategorie „Kredit“ zugeordnet ist, wird der abstrakte Prozessbaustein „Kreditrisiko ermitteln“ zugeordnet, der durch eine Generalisierungs- und Spezialisierungsbeziehung mit den spezielleren Prozessbausteinen verknüpft ist.

Rupprecht schlägt die Verwendung von Namenskonventionen für die Bezeichnung von Prozesskategorien, Prozessbausteinen und Musterprozessen vor, um diese zu vereinheitlichen und das Speichern von gleichen oder ähnlichen Objekten zu vermeiden. Er schlägt jedoch keine Namenskonventionen vor. [Rupp02, 87] Zwischen Prozessbausteinen oder Musterprozessen sind keine Überprüfungen auf Übereinstimmung vorgesehen (beispielsweise gleiche oder ähnliche Prozessschritte, Namen, etc.). Werden Prozessbausteine oder Musterprozesse auf Basis bereits vorhandener Prozessbausteine oder Musterprozesse definiert, wird in ein neu erzeugtes Objekt eine Kopie eingefügt. Dieses soll *„unter einem anderen Namen“* abgelegt werden [Rupp02, 95f].

**Bibliothek für generische Rahmenbedingungen**

Die Rahmenbedingungen werden semi-formal beschrieben, da die *„Einflussgrößen und Ausprägungen“* formal getrennt werden, die Bezeichnung dieser wird jedoch den Benutzern überlassen [Rupp02, 78].

Die Bibliothek für die Rahmenbedingungen besteht ebenfalls aus einer Ordnerhierarchie. Rupprecht schlägt eine Gliederung in Anlehnung an verschiedene Autoren (z.B. [Küpp82, 80]) in unternehmensinterne und -externe Einflussgrößen vor [Rupp02, 32]. Die Unterteilung in weitere Kategorien kann aus dem Gliederungsschema in der Arbeit von Rupprecht entnommen werden. Der Vorschlag der Gliederung bezieht sich auf die oberen zwei Ebenen der Ordnerstruktur und kann durch die Benutzer erweitert werden. Die Erweiterung kann sowohl auf der gleichen Ebene durchgeführt werden, als auch auf den unteren Ebenen. [Rupp02, 37]

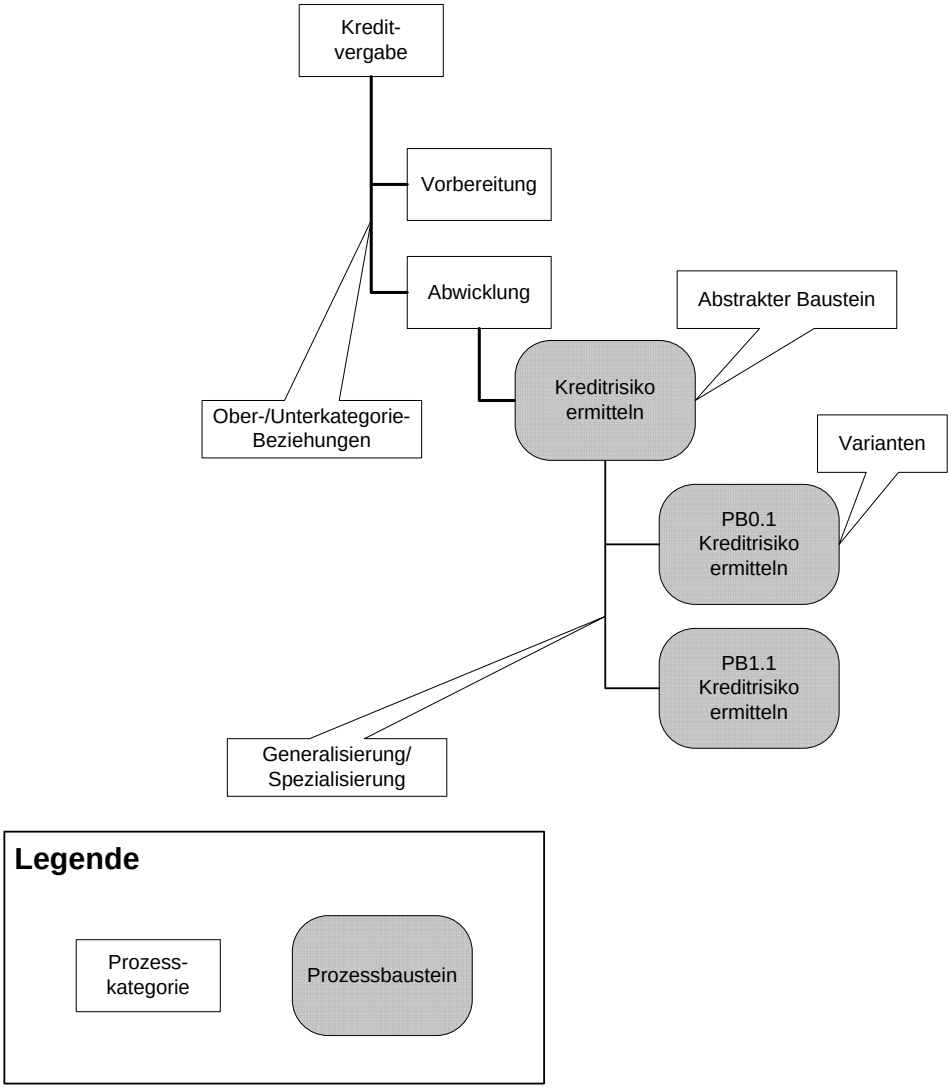



**Abbildung 84: Beispiel einer Ordnerstruktur für Prozessbausteine in Anlehnung an [Rupp02, 87]**

Abbildung 85 zeigt ein Beispiel für die Ordnerstruktur generischer Rahmenbedingungen. Es wurde ein Ausschnitt aus dem vorgeschlagenen Gliederungsschema ausgewählt. Den Kategorien der Einflussgrößen werden Einflussgrößen zugeordnet. Diese können wieder mehreren Einflussgrößenkategorien zugeordnet werden. Den Einflussgrößen werden anschließend die möglichen Ausprägungen zugeordnet. [RuPR99, 7f]

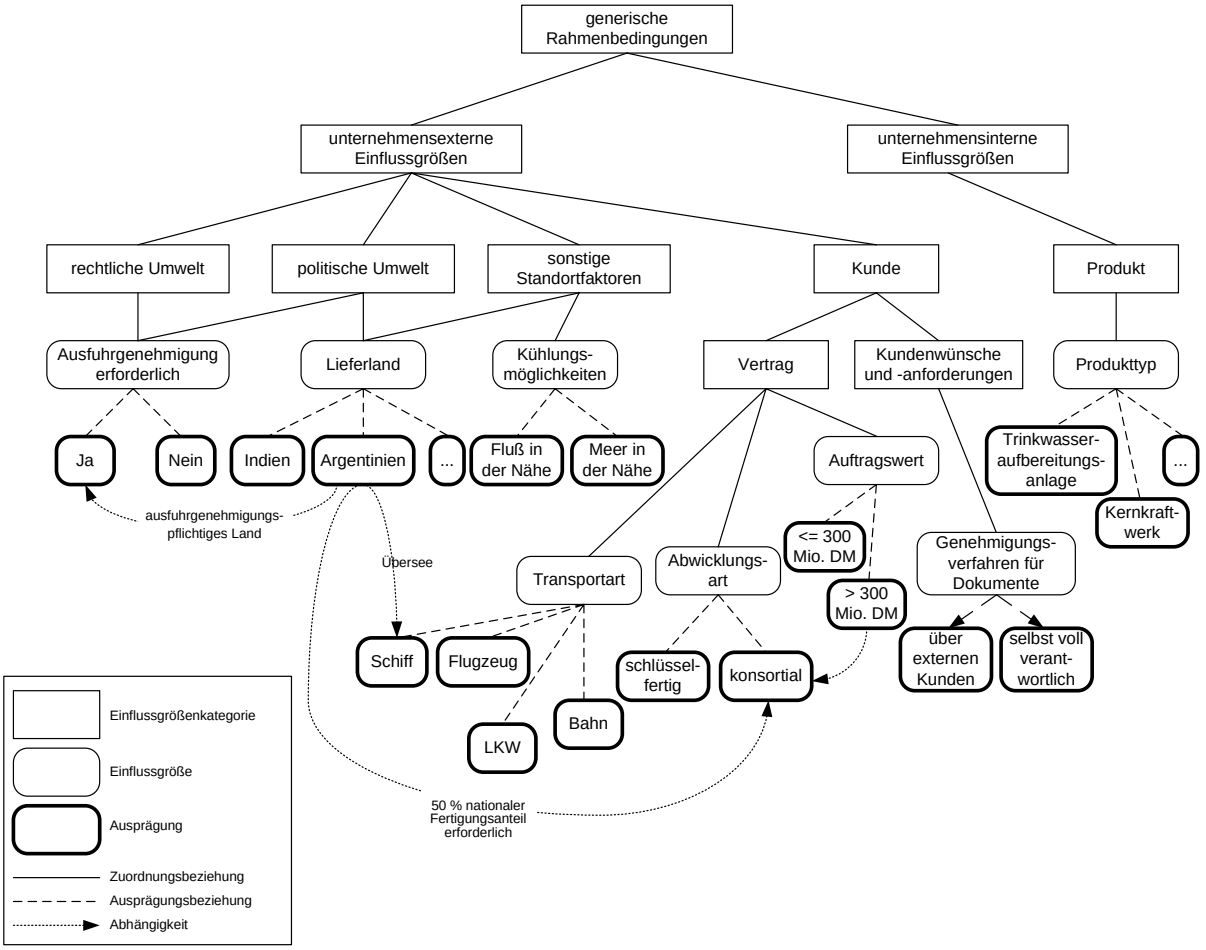



**Abbildung 85: Beispiel für Ordnerstruktur generischer Rahmenbedingungen [RuPR99, 8]**

Rupprecht hat sein Konzept der Prozessindividualisierung durch einen unvollständigen Prototypen realisiert. Dieser Prototyp wird dem horizontalen Prototyping zugeordnet. [Rupp02, 8] Dabei werden *„alle anvisierten Systemfunktionen in vereinfachter Form realisiert“* [Floy92]. Abbildung 86 zeigt die am häufigsten verwendeten Dialogfenster des Prototyps mit einem Aufrufdiagramm. Einstiegspunkt des Prototyps ist der Prozess-Editor, in dem die Prozessmodellierung durchgeführt wird [Rupp02, 150]. Zur Realisierung des Prozess-Editors wurde POWM (Prozessorientiertes Wissensmanagement) als Modellierungswerkzeug und -methode verwendet. Es können jedoch auch andere Modellierungsmethoden und -werkzeuge verwendet werden, sofern sie die Basisfunktionalitäten von POWM umsetzen. [Rupp02, 8] Dies sind Funktionalitäten wie verteilte Prozessmodellierung, eine Benutzerschnittstelle zur Modellierung von Prozessmodellen und Unterstützung der Dekomposition von Prozessmodellen [Rupp02, 130].

Auf die einzelnen Dialogfenster wird hier nicht näher eingegangen, da dies die Bewertung nicht beeinflusst und daher nicht Gegenstand dieser Diplomarbeit ist.

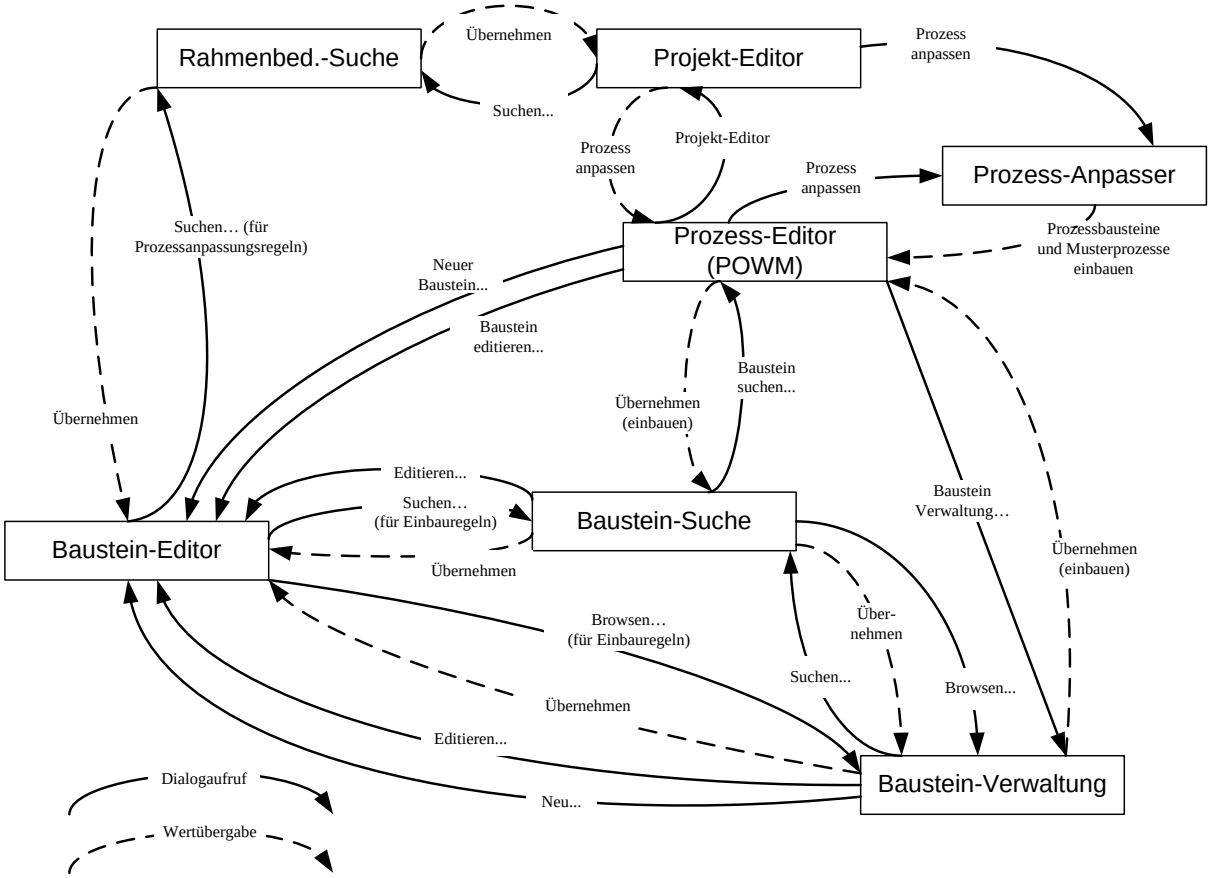



**Abbildung 86: Dialogfenster mit Aufrufdiagramm [Rupp02, 150]**

## *4.3 Wiederverwendung anhand des Rahmenbeispiels*

Die Wiederverwendung von Prozessmodellen bzw. Teilbereichen daraus wird beispielhaft in Abbildung 87 durch den *„projektspezifischen Einbau eines Prozessbausteins“* dargestellt [RuPR99, 13ff]. Die projektspezifischen Rahmenbedingungen werden in der Ordnerstruktur generischer Rahmenbedingungen gesucht. Zu der Einflussgröße „Lieferland“ wird die Ausprägung „Argentinien“ gewählt. Die Rahmenbedingung „Lieferland: Argentinien“ verfügt über eine Kontextgestaltungsregel zu der Rahmenbedingung „Ausfuhrgenehmigung: Ja“ (1), die ebenfalls hinzugefügt wird. Die ausgewählten Rahmenbedingungen werden in ein *„Modell von Projektrahmenbedingungen“* zusammengeführt (2). Für jede Rahmenbedingung, werden dem Benutzer verknüpfte Prozessbausteine oder Musterprozesse durch die Prozessanpassungsregel vorgeschlagen, die er in diesem Fall hinzufügen kann oder nicht (3). Für jeden akzeptierten Prozessbaustein wird überprüft, ob eine Einbauregel zu anderen Prozessbausteinen existiert. In diesem Beispiel wurde die Einbauregel *„ist Vorgänger von“* definiert, und somit ein Vorschlag über das Einfügen des nachfolgenden Prozessbausteins generiert (4) und in *„die projektspezifische Prozessmodellinstanz“* eingefügt werden kann (5). [Rupp02, 68]

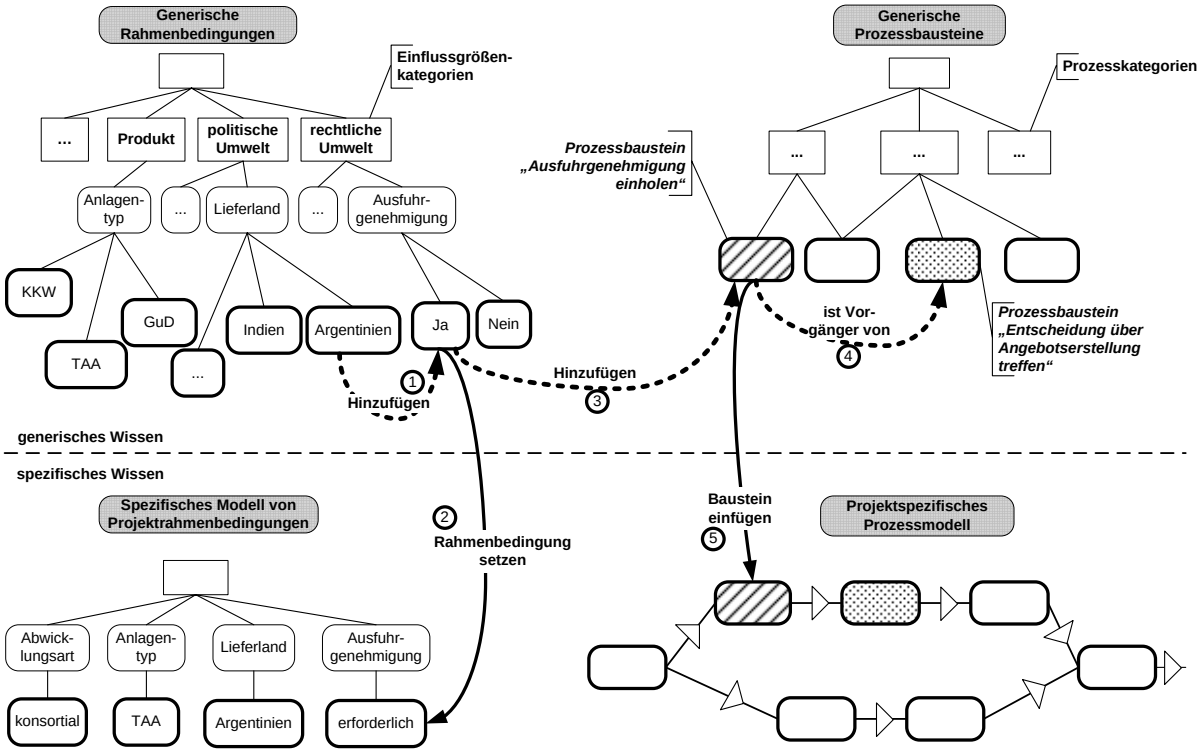



**Abbildung 87: Projektspezifischer Einbau eines Prozessbausteins [RuPR99, 14]**

Das Vorgehensmodell aus Abbildung 88 stellt den Ablauf der reaktiven Wiederverwendung nach diesem Ansatz dar [Rupp02, 123ff]. Dazu wird auf die Funktionen der Wissenserwerbskomponente, der Faktenbeschreibungskomponente und der Problemlösungs- und Erklärungskomponente zurückgegriffen und das Rahmenbeispiel darauf angewendet. Auf eine Zuordnung der Funktionen zu der jeweiligen Komponente wird verzichtet. Anschließend erfolgt eine Zuordnung der einzelnen Schritte zu der in Abschnitt 2.2.2.1 eingeführten Einteilung.

Zuerst werden in der Projektinitialphase auf der obersten Ebene der Prozesshierarchie Rahmenbedingungen gewählt, wobei die benötigten Ausprägungen ausgewählt werden (Schritt 1). Dazu wird die Funktion „Projektrahmenbedingungen setzen“ verwendet. Im nächsten Schritt erfolgt eine Prüfung mit Hilfe von CBR-Methoden (Methoden des Case Based Reasonings), ob zu diesen Rahmenbedingungen bereits ein gleicher oder ähnlicher Prozessfall vorhanden ist (Schritt 2). Wird ein gleicher oder ähnlicher Prozessfall gefunden, kann er kopiert und wiederverwendet werden (Schritt 3). In Schritt 4 können die bereits ausgewählten Projektrahmenbedingungen gegebenenfalls manuell angepasst werden. Ist kein Prozessfall vorhanden, wird geprüft, ob ein Musterprozess zu diesen Rahmenbedingungen vorhanden ist (Schritt 5). Wird ein Musterprozess gefunden, können sie mit der Funktion „Musterprozess einbauen“ in den aktuellen Prozessfall eingebaut werden (Schritt 6). In Schritt 7 können wiederum Projektrahmenbedingungen ergänzt werden. Wird auch kein Musterprozess gefunden, muss eine Prozesshierarchie *„manuell mit Hilfe eines Prozessmodellierungswerkzeugs erstellt werden“* (Schritt 8). In Schritt 9 können Projektrahmenbedingungen den einzelnen Teilprozessen der Prozesshierarchie *„auf verschiedenen Ebenen zugeordnet werden“*. Dazu wird wieder die Funktion „Projektrahmenbedingungen setzen“ verwendet. Durch die Rückkopplung zu Schritt 2 kann nun erneut nach einem bereits vorhandenen Prozessfall bzw. Musterprozess gesucht werden. In Schritt 10 *„werden die Prozessmodellinstanzen an die geänderten und neu hinzugekommenen Projektrahmenbedingungen angepasst“*. [Rupp02, 123ff]

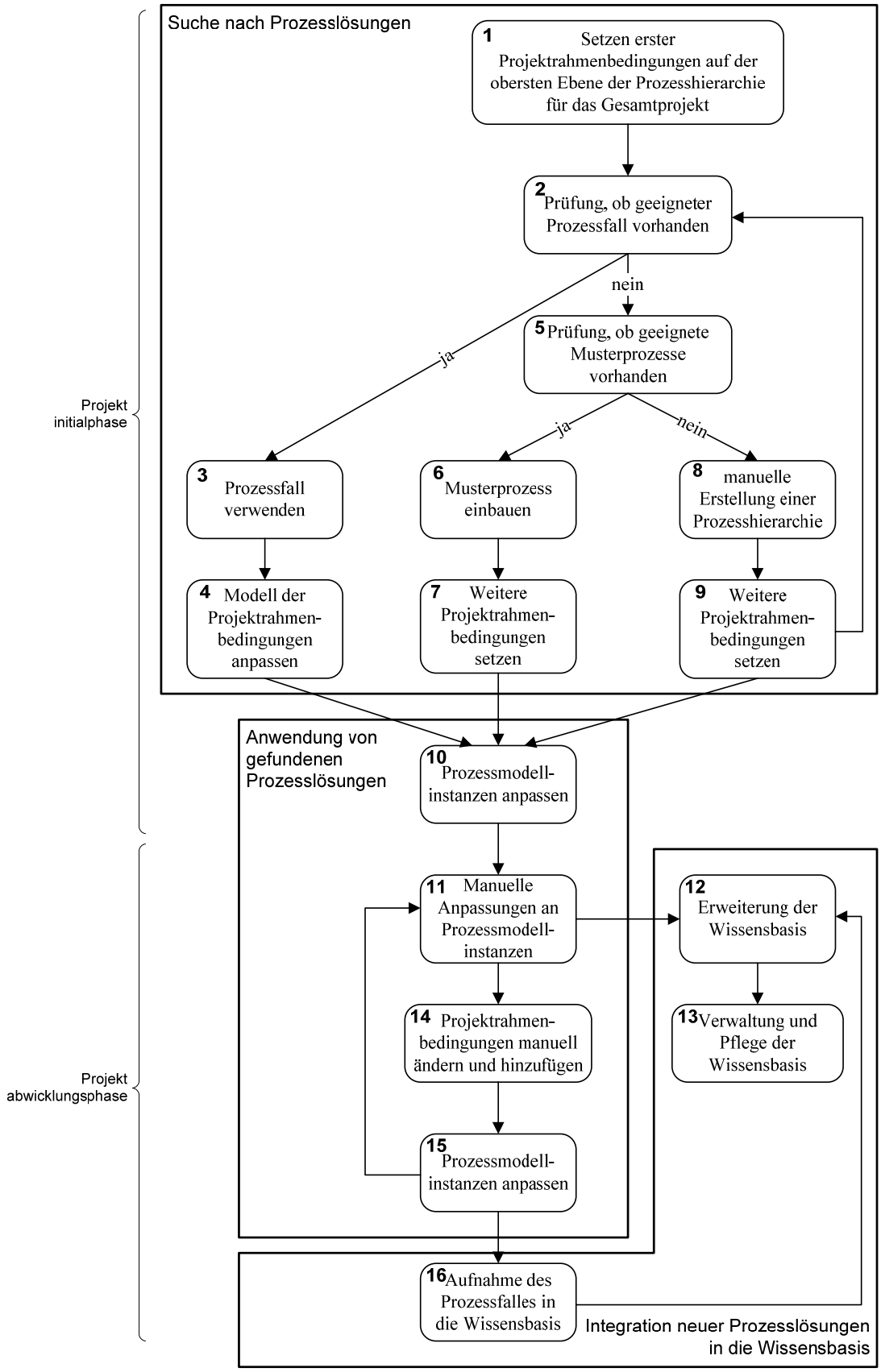



**Abbildung 88: Vorgehensmodell zur Wiederverwendung in Anlehnung an [Rupp02, 124]**

In der *Projektabwicklungsphase* werden in Schritt 10 die gemäß der gewählten Rahmenbedingungen abhängigen Prozessbausteine oder Musterprozesse eingebaut. Anschließend werden iterativ die Prozessmodellinstanzen mit Hilfe des

Prozessmodellierungswerkzeugs manuell angepasst (Schritt 11), die Projektrahmenbedingungen manuell geändert und hinzugefügt (Schritt 14) und neuerlich die Prozessbausteine oder Musterprozesse eingebaut (Schritt 15). Werden neue Prozessbausteine, Musterprozesse bzw. neue generische Rahmenbedingungen samt ihren Gestaltungsregeln aufgedeckt, können diese in die Wissensbasis aufgenommen werden (Schritt 12). Die Wissensbasis muss auch regelmäßig verwaltet und gepflegt werden (Schritt 13). Zu guter Letzt wird der soeben erstellte Prozessfall in die Wissensbasis aufgenommen (Schritt 16). [Rupp02, 122f]

Aus diesem Vorgehensmodell ist ersichtlich, dass auch bei diesem Ansatz von einer bereits existierenden Wissensbasis ausgegangen wird. Im Zuge der Untersuchung wird zuerst der Abschnitt 4.3.1 (*Integration von externen Prozesslösungen*) für die Untersuchung eingeführt, bei der die Prozessbausteine, Rahmenbedingungen und die Gestaltungsregeln definiert werden, um das generische Wissen für die anschließende reaktive Wiederverwendung explizit zu machen. Die Schritte aus dem Vorgehensmodell werden folgender Einteilung zugeordnet: *Suche nach geeigneten Prozesslösungen* (Abschnitt 4.3.2), *Anwendung von gefundenen Prozesslösungen* (Abschnitt 4.3.3), *Überprüfung der angepassten Prozesslösungen* (Abschnitt 4.3.4) und *Integration neuer Prozesslösungen in die Wissensbasis* (Abschnitt 4.3.5). Nachstehende Tabelle zeigt eine Zuordnung der Schritte aus dem Vorgehensmodell zu den Einteilungen.

| **Einteilung** | **Rupprecht** |
|---|---|
| Integration von externen Prozesslösungen | - |
| Suche nach geeigneten Prozesslösungen | Schritte 1, 2, 3, 4, 5, 6, 7, 8 und 9 |
| Anwendung von gefundenen Prozesslösungen | Schritte 10, 11, 14, 15 |
| Überprüfung der angepassten Prozesslösungen | Schritt 10, 15 |
| Integration neuer Prozesslösungen in die Wissensbasis | Schritte 12, 13, 16 |

**Tabelle 6: Zuordnung der gemeinsamen Konzepte und Aufgaben nach Rupprecht**

### 4.3.1 Integration von externen Prozesslösungen

Grundlage für diesen Ansatz ist die Wissensbasis, die durch die Wissenserwerbskomponente gestaltet wird. Es werden generische *„Prozessmodelle, Rahmenbedingungen und Gestaltungsregeln“* definiert und verwaltet [Rupp02, 116]. Die generischen Prozessmodelle bestehen aus Prozessbausteinen und Musterprozessen, die *„weitere generische Prozessmodelle“* und Prozessbausteine enthalten und somit adaptive Referenzprozesse darstellen [Rupp02, 75].

Funktionen wie „Prozessbaustein definieren“, „Musterprozess definieren“ und „generische Rahmenbedingungen definieren“ werden für den Wissenserwerb verwendet. Die Gestaltungsregeln, die die eigentliche Wissensrepräsentation vornehmen, werden durch die Funktionen „Kontextgestaltungsregel definieren“, „Prozessanpassungsregel definieren“ und „Einbauregel definieren“ erstellt. [Rupp02, 93ff]

Die Suche nach und Verwaltung von generischen Prozessmodellen und Rahmenbedingungen erfolgt ebenfalls in der Wissenserwerbskomponente, und es wird auf die Funktionen „generische Rahmenbedingungen verwalten“, „generische Rahmenbedingung suchen“,

„generische Prozessmodelle verwalten“ und „generisches Prozessmodell suchen zurückgegriffen [Rupp02, 102ff]. Dies dient der *„Verwaltung und Pflege der Wissensbasis“*, die laufend durchgeführt werden soll [Rupp02, 123f], ist jedoch nicht Gegenstand der Untersuchung.

***Funktion „Prozessbaustein definieren“***

Um einen Prozessbaustein zu definieren, wird ein Ausschnitt aus der Prozessmodellinstanz, welche durch Elemente der Prozesslösung mit ihren Beziehungen dargestellt werden, oder aus einem vorhandenen Prozessbaustein ausgewählt. Dazu wird die *„Ordnerstruktur für Prozessbausteine“* durchsucht. Der Prozessbaustein kann jedoch auch komplett neu erstellt werden (*„auf der grünen Wiese“*). Wird der Prozessbaustein nicht neu erstellt, wird eine Kopie erzeugt und ein neues Objekt angelegt, das den Prozessbaustein darstellen soll. [Rupp02, 94f]

Im nächsten Schritt müssen Attributwerte vergeben werden, wobei unbedingt der Name, Erfahrungswert und Qualitätsstatus erforderlich sind. Ausprägungen von Erfahrungswerten sind „Best Practice“ und „Lessons Learned“ und bezeichnen somit positive und negative Erfahrungen. Ausprägungen von Qualitätsstatus sind beispielsweise „Entwurf“, „freigegeben“, „abgestimmt“ oder „Qualitätsbaustein“ und kann bei „Best-Practice-Bausteinen“ vergeben werden. [Rupp02, 83ff]

Der soeben erstellte Prozessbaustein wird in das Ordnungsraster für generische Prozessmodelle eingeordnet (Bibliothek für generische Prozessmodelle), wobei er entweder einer Prozesskategorie oder als Spezialisierung, das heißt bei diesem Ansatz als Variante, *„eines anderen Prozessbausteins“* zugeordnet werden kann. Am Ende dieser Funktion wird der soeben definierte Prozessbaustein in die Ordnerstruktur gegebenenfalls in mehreren Ordnern redundant gespeichert [Rupp02, 95].

Zur Verwaltung von generischen Prozessmodellen kann aus dieser Funktion die Funktion „generische Prozessmodelle verwalten“ aufgerufen werden. Mit Hilfe dieser Funktion können neue Prozesskategorien angelegt werden, neue Prozessbausteine bzw. Musterprozesse definiert werden, diese umbenannt, in der Ordnerstruktur verschoben, gelöscht oder bearbeitet werden. [Rupp02, 95, 103]

***Beispiel 4.3.1.1***

> *Für den ersten Prozessbaustein wird aus dem Prozessmodell model_100 der Ausschnitt „Kreditrisiko ermitteln“ ausgewählt und kopiert (analog zu P0.1 aus Hagen). Für den Prozessbaustein wird ein neues Objekt angelegt.*

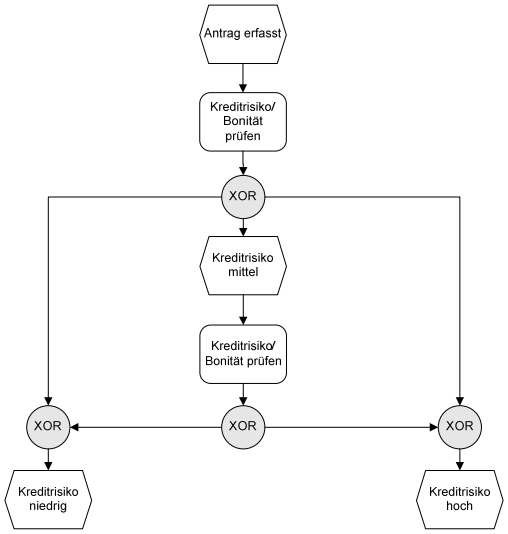



**Abbildung 89: Prozessbaustein „PB0.1 Kreditrisiko ermitteln“**

*Dem Prozessbaustein „PB0.1 Kreditrisiko ermitteln“ wird der Attributwert Name vergeben. Die restlichen Attribute (Erfahrungswert und Qualitätsstatus) werden hier vernachlässigt, da sie für die Untersuchung keinen Mehrwert liefern.*

*Der Prozessbaustein „PB 0.1 Kreditrisiko ermitteln“ wird nun in eine vorhandene Ordnerstruktur eingeordnet (auf eine vorherige Erstellung der Ordnerstruktur wird hier verzichtet).*

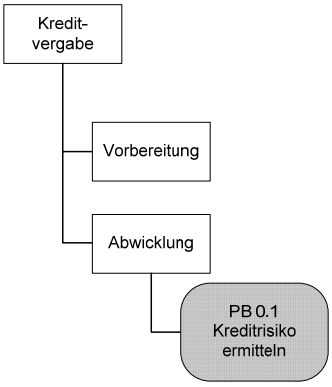



**Abbildung 90: Einordnung des Prozessbausteins in Ordnerstruktur generischer Prozessmodelle**

*Dieselbe Vorgehensweise wird mit den Ausschnitten „Kreditantrag ablehnen“ (Abbildung 91) und „Kreditantrag bewilligen“ (Abbildung 92) durchgeführt.*

*Diese Prozessbausteine werden ebenfalls in die vorhandene Ordnerstruktur eingeordnet (siehe Abbildung 93).*

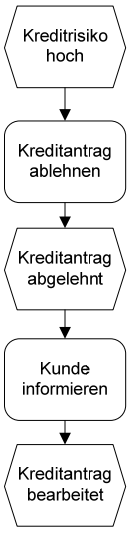



**Abbildung 91: Prozessbaustein „PB0.2 Kreditantrag ablehnen“**

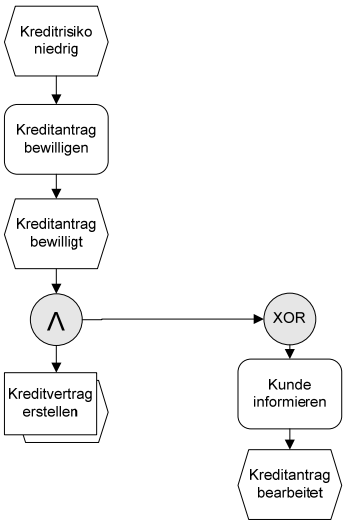



**Abbildung 92: Prozessbaustein „PB0.3 Kreditantrag bewilligen“**

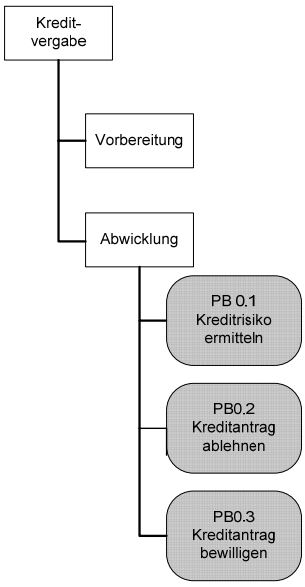



**Abbildung 93: Ordnerstruktur generischer Prozessmodelle von model_100[3]**

[3] Die Darstellungen der Ordnerstruktur der Prozessbausteine erfolgt in Anlehnung an das Beispiel aus Rupprecht (vgl. Abbildung 84).

*Dieselbe Vorgehensweise wird auch mit den Teilprozessen des Prozessmodells model_101 aus Abschnitt 1.4 durchgeführt. Die Teilprozesse werden in Anlehnung an die Prozessmuster aus dem Ansatz von Hagen definiert. Auf eine erneute Darstellung der Prozessschritte wird verzichtet.*

*Die beiden Prozessmodelle model_100 und model_101 stellen Varianten dar. Es wird ein abstrakter Baustein vor den jeweiligen Prozessbaustein eingeführt, wobei die Varianten Spezialisierungen des abstrakten Bausteins sind. Es ergibt sich die in Abbildung 94 dargestellte Ordnerstruktur.*

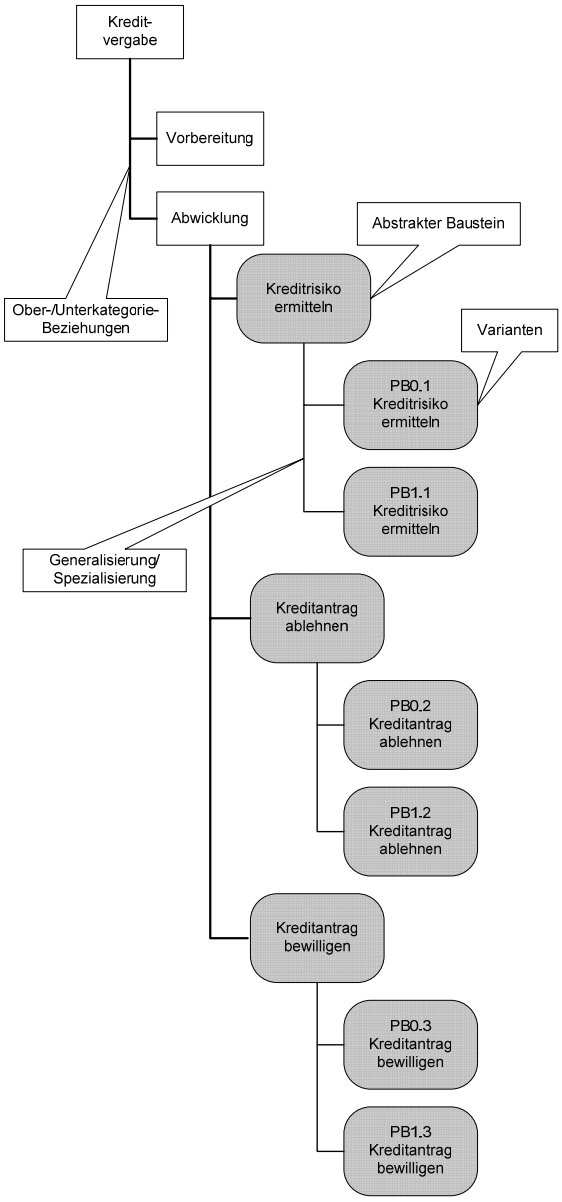



**Abbildung 94: Ordnerstruktur generischer Prozessmodelle von model_100 und model_101**

*Die gleiche Vorgehensweise wird für die Dokumentation der Prozessbausteine aus den Prozessmodellen model_102, model_1021 und model_1022 durchgeführt. Die Ausschnitte, die den Prozessmustern von Hagen entsprechen (P3, P3.2, P3.3, P3.2.1 und P3.3.1), werden nicht als Prozessbausteine definiert, da Prozessbausteine keine weiteren Prozessbausteine enthalten dürfen.*

*Daher werden der Prozessbaustein „PB3.1 Kredit beantragen“ (entspricht dem Prozessmuster „P3.1 Kredit beantragen“, Seite 78) und die abstrakten Bausteine „Unterlagen beschaffen“ und „Kredithöhe ermitteln“ der Prozesskategorie „Vorbereitung“ zugeordnet. Die abstrakten Bausteine werden erstellt, da diesem unterschiedliche Varianten zugeordnet werden. „PB3.2.1.1 Unterlagen beschaffen“ (entspricht „P3.2.1.1 Unterlagen beschaffen“, Seite 81) und „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“ (entspricht „P3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“, Seite 87) werden als Varianten dem abstrakten Baustein „Unterlagen beschaffen“ zugeordnet. „PB3.2.1.2 Kredithöhe anhand der Unterlagen für Privatkredit ermitteln“ (entspricht „P3.2.1.2 Kredithöhe anhand der Unterlagen für Privatkredit ermitteln“, Seite 82) und „PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“ (entspricht „P3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“, Seite 88) werden dem abstrakten Baustein „Kredithöhe ermitteln“ zugeordnet.*

*Aus Gründen der Übersichtlichkeit wird auf eine nochmalige Darstellung der Prozessmodelle model_100 und model_101 in der Ordnerstruktur unter der Kategorie „Abwicklung“ verzichtet.*

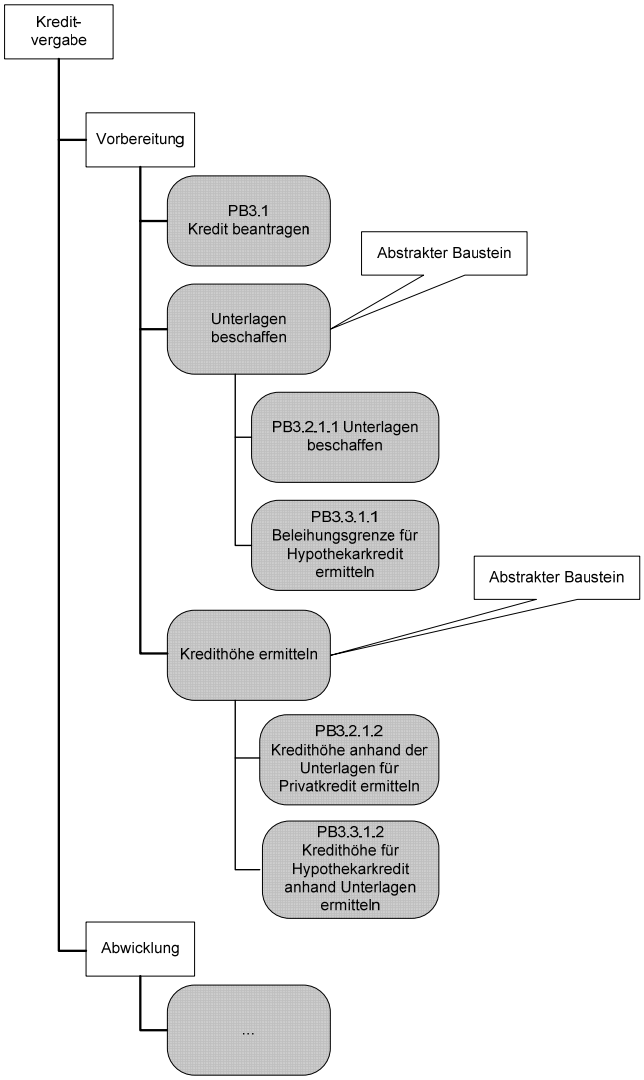



**Abbildung 95: Ordnerstruktur generischer Prozessmodelle von model_102, 1021 und 1022**

Eine alternative Darstellungsweise einer Ordnerstruktur generischer Prozessmodelle zeigt folgende Abbildung:

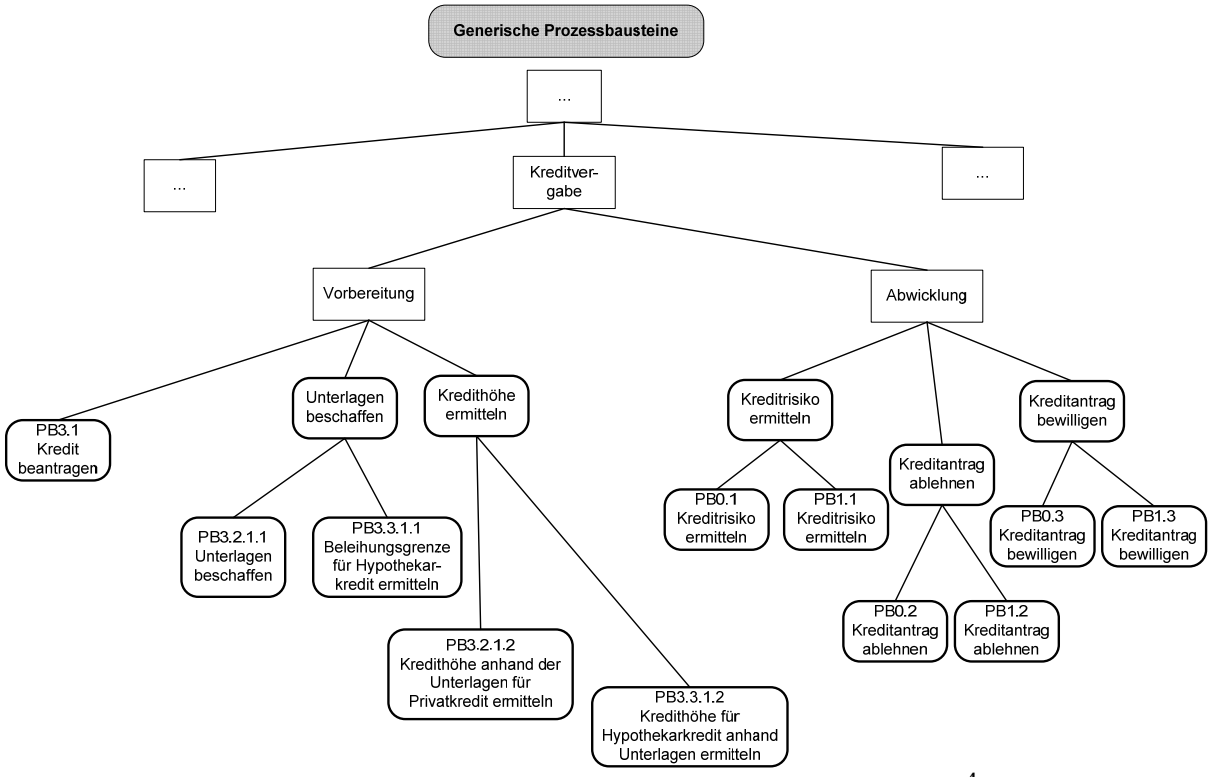



**Abbildung 96: Alternative Ordnerstruktur für Prozessbausteine**[4]

Aus der Funktion „Prozessbaustein definieren“ können folgende Funktionen aufgerufen werden: „Generische Rahmenbedingungen definieren“, „Kontextgestaltungsregel definieren“, „Prozessanpassungsregel definieren“, und „Einbauregel definieren“. [Rupp02, 95] Auf diese Funktionen wird in nachfolgenden Ausführungen eingegangen. Sie werden benötigt, um zu den definierten Prozessbausteinen relevante Rahmenbedingungen, die sich in Einflussgrößen und Ausprägungen unterteilen, zwischen den Rahmenbedingungen, zwischen den Rahmenbedingungen und Prozessbausteinen oder Musterprozessen und zwischen den Prozessbausteinen Abhängigkeiten zu definieren.

Doch bevor zu den definierten Prozessbausteinen die Rahmenbedingungen und Regeln definiert werden, wird noch ein Musterprozess definiert.

### *Funktion „Musterprozess definieren“*

Dieser Ablauf ist weitgehend ident mit der Funktion „Prozessbaustein definieren“. Lediglich variable Prozessbausteine müssen zusätzlich eingebaut werden. Desweiteren ist es nicht möglich, Einbauregeln für Musterprozesse zu definieren (Aufruf der Funktion „Einbauregel definieren“ ist aus dieser Funktion nicht möglich). Da ein Musterprozess zu Beginn alle optionalen Prozessbausteine enthält, ist die Prozessanpassungsregel vom Aktionstyp „Entfernen“ besonders wichtig. [Rupp02, 90, 96] Somit können einzelne Prozessbausteine bei Bedarf entfernt werden.

[4] Diese Ordnerstruktur wurde in Anlehnung an Rupprecht erstellt (vgl. [Rupp02, 68]).

***Beispiel 4.3.1.2***

*Für das Prozessmodell model_1022 aus Abschnitt 1.4 wird ein Musterprozess definiert. Es werden alle Funktionen und Ereignisse samt ihren Konnektoren ausgewählt (ohne Prozesswegweiser) und der Attributwert Name „MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit“ vergeben. Die restlichen Attribute (Erfahrungswert und Qualitätsstatus) werden hier in Analogie zu Beispiel 4.3.1.1 auf Seite 115 ebenfalls vernachlässigt. Er wird in die Kategorie „Vorbereitung“ der Ordnerstruktur für Prozessmodelle eingefügt. Auf eine grafische Darstellung der Einordnung in die Ordnerstruktur wird hier verzichtet. Die eingebauten Prozessbausteine zu diesem Musterprozess sind die Prozessbausteine „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“ und „PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“. Folgende Darstellung zeigt den Musterprozess mit den variablen Prozessbausteinen (die zu den Prozessbausteinen zugehörigen Prozessschritte sind gekennzeichnet).*

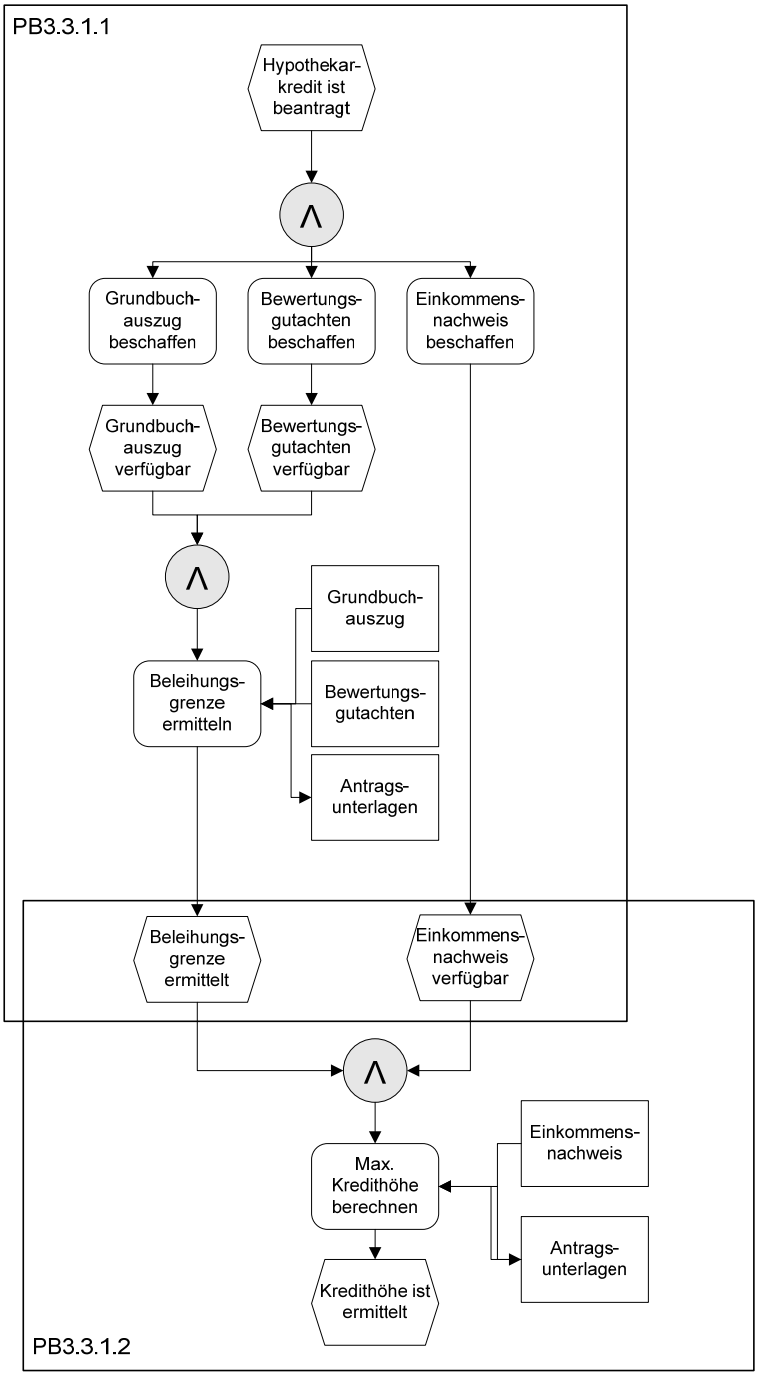



**Abbildung 97: Musterprozess MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit**

### *Funktion „Generische Rahmenbedingungen definieren“*

Um generische Rahmenbedingungen zu definieren, muss eine neue Einflussgröße definiert werden. Dazu wird Name und Art („boolean“, „diskret“ oder „nummerisch“) der Einflussgröße vergeben. Für die Einflussgröße wird ein neues Objekt angelegt und eine Zuordnungsbeziehung zu der ausgewählten Einflussgrößenkategorie angelegt. Ist die Art „boolean“ werden automatisch die Ausprägungen „Ja“ und „Nein“ angelegt. Bei der Art „diskret“ können für Ausprägungen Texte eingegeben werden, und nummerische Ausprägungen enthalten Werteinheiten. [Rupp02, 97]

***Beispiel 4.3.1.3***

*Zu den Prozessmodellen model_100 und model_101 werden folgende Einflussgrößen mit ihren Ausprägungen definiert:*

1. *Einflussgröße zu PB0.1 (Abbildung 8989): Name „**Kreditrisiko**“ und Art „boolean“ wird der Kategorie „Bankvorschriften“ zugeordnet.*

2. *Einflussgrößen zu PB0.2 und zu PB0.3 werden im Rahmen dieser Diplomarbeit nicht benötigt, da sie über den Prozessbaustein PB0.1 angesprochen werden können (vgl. weiter unten die Einbauregeln).*

*Für das model_101 müssen keine zusätzlichen Einflussgrößen definiert werden, da es eine Variante zu model_100 darstellt.*

*Zu den Prozessmodellen model_102, model_1021 und model_1022 werden folgende Einflussgrößen mit ihren Ausprägungen definiert:*

1. *Einflussgröße zu PB3.1: wird nicht benötigt, da es sich bei dieser Prozesslösung um eine Unterscheidung zwischen Privat- und Hypothekarkredit handelt (vgl. Abbildung 54, Seite 78) und daher nicht angelegt.*

2. *Einflussgröße zu PB3.2.1.1, PB3.3.1.1 und MP3.3.1: Name „**Unterlagen**“ und Art „diskret“ und wird der Kategorie „Bankvorschriften“ zugeordnet.*

3. *Einflussgröße zu PB3.2.1.2, PB3.3.1.2 und MP3.3.1: Name „**Kredithöhe**“ und Art „diskret“ und wird der Kategorie „Bankvorschriften“ zugeordnet.*

*Eine weitere Einflussgröße mit ihrer Ausprägung ist: Name „**Kredit**“ und Art „diskret“. Diese wird der Kategorie „Produkt“ zugeordnet und wird in diesem Beispiel eingeführt, um die Abhängigkeiten zwischen Rahmenbedingungen „Unterlagen“ und „Kredithöhe“ zu definieren. Es ist nur eine Möglichkeit der Umsetzung. Abbildung 98 stellt die soeben erstellten Einflussgrößen dar.*

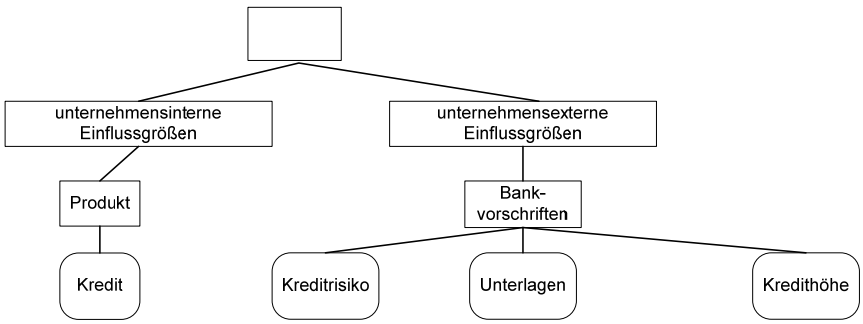



**Abbildung 98: Einflussgrößen und Ausprägungen**

*Einen booleschen Wert hat die Einflussgröße „Kreditrisiko“, daher werden die Ausprägungen automatisch hinzugefügt (vgl. Abbildung 99).*

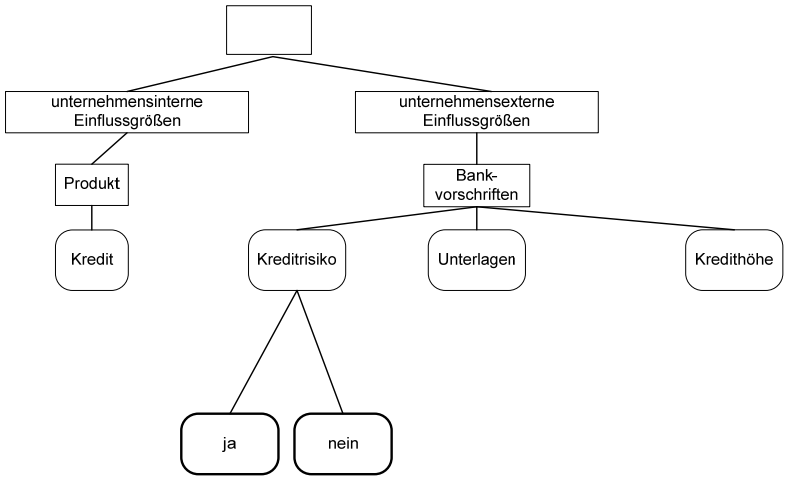



**Abbildung 99: Einflussgrößen und Ausprägungen**

*Die restlichen Einflussgrößen werden als diskret definiert und haben textuelle Ausprägungen. Zu jeder Einflussgröße werden in diesem Beispiel die Ausprägungen „Privatkredit“ und „Hypothekarkredit“ hinzugefügt. Es ergibt sich somit folgende Ordnerstruktur für die generischen Rahmenbedingungen.*

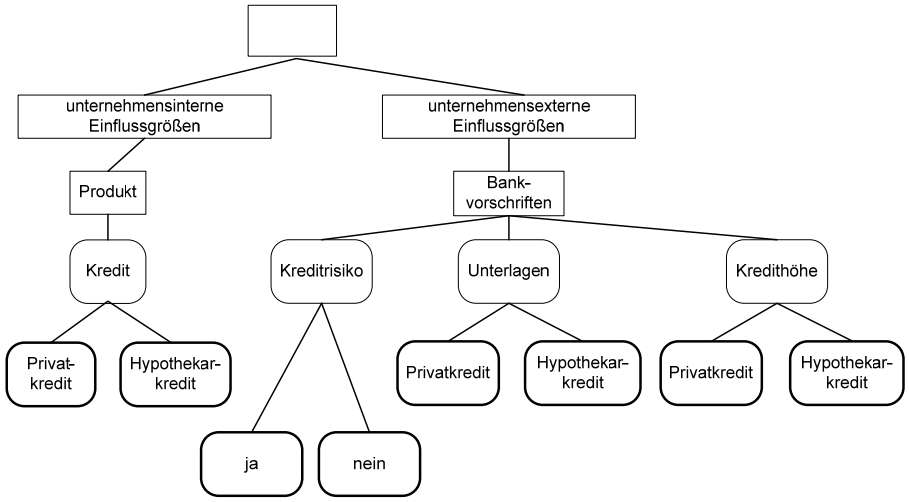



**Abbildung 100: Einflussgrößen und Ausprägungen**

Diese Vorgehensweise geschieht nicht wie vorgestellt auf einmal, sondern für jede Einflussgröße iterativ, da mit Hilfe dieser Funktion nur eine Einflussgröße definiert werden kann [Rupp02, 97].

### *Funktion „Kontextgestaltungsregel definieren“*

Mit Hilfe der Kontextgestaltungsregeln werden Abhängigkeiten zwischen Rahmenbedingungen definiert. So können zusätzliche Rahmenbedingungen ausgewählt werden. Auch hier kann pro Aufruf der Funktion eine Kontextgestaltungsregel definiert werden. Dazu müssen eine oder mehrere Rahmenbedingungen aus der *„Ordnerstruktur für Rahmenbedingungen“* ausgewählt werden und als Prämissen definiert werden. Die selektierten Ausprägungen werden mit einer weiteren Rahmenbedingung verknüpft, die als Konklusion definiert wird. Die Kontextgestaltungsregeln sind stets vom Typ *Hinzufügen*. Wird diese Funktion aus der Funktion „Prozessbaustein definieren“ heraus aufgerufen, werden die Prämissen (Rahmenbedingungen) bereits als Parameter mitgeliefert. [Rupp02, 99]

***Beispiel 4.3.1.4***

*Als Rahmenbedingung wird die Ausprägung „Hypothekarkredit“ der Einflussgröße „Kredit“ als Prämisse gewählt. Als Konklusion wird jeweils die Ausprägung „Hypothekarkredit“ bei den Einflussgrößen „Unterlagen“ und „Kredithöhe“ gewählt. Grafisch können die Kontextgestaltungsregeln wie in Abbildung 101 dargestellt werden.*

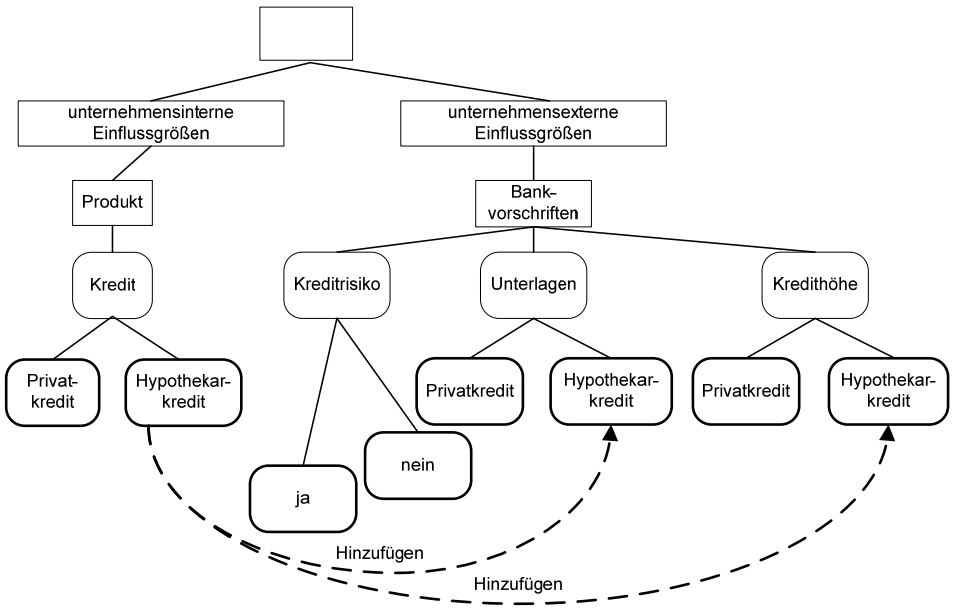



**Abbildung 101: Generische Rahmenbedingungen mit Kontextgestaltungsregeln**

### *Funktion „Prozessanpassungsregel definieren“*

Die Prozessanpassungsregeln dienen im weiteren Verlauf dazu, dem Benutzer ausgehend von den gesetzten Rahmenbedingungen generische Prozessmodelle (Prozessbausteine oder Musterprozesse) als Vorschlag zu liefern [Rupp02, 119].

Zur Definition können ebenfalls eine oder mehrere Rahmenbedingungen ausgewählt werden, die die Prämissen sind. Als Konklusion wird ein Musterprozess oder Prozessbaustein ausgewählt. Wird diese Funktion aus der Funktion „Prozessbaustein definieren“ oder „Musterprozess definieren“ heraus aufgerufen, entfällt die Auswahl des Prozessbausteins bzw. Musterprozesses. Zu guter Letzt wird der Abhängigkeitstyp bestimmt, wobei zwischen „Hinzufügen“ oder „Entfernen“ unterschieden wird. [Rupp02, 100]

***Beispiel 4.3.1.5***

*Folgende Rahmenbedingungen werden als **Prämisse** ausgewählt. Dabei wird wiederum für jede Prozessanpassungsregel die Funktion erneut aufgerufen.*

*a) Auswahl „Kreditrisiko: ja“*
*b) Auswahl „Unterlagen: Hypothekarkredit“*
*c) Auswahl „Kredithöhe: Hypothekarkredit“*
*d) Auswahl „Unterlagen: Hypothekarkredit“ UND „Kredithöhe: Hypothekarkredit“*

*Folgende Prozessbausteine und folgender Musterprozess werden zu den jeweiligen Rahmenbedingungen als* ***Konklusion*** *ausgewählt (dabei wird a mit a verknüpft usw.)*

*a) Auswahl „PB0.1 Kreditrisiko ermitteln“*
*b) Auswahl „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“*
*c) Auswahl „PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“*
*d) Auswahl „MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit“*

*Alle vier Prozessanpassungsregeln sind vom Aktionstyp „Hinzufügen“.*

*Grafisch wird es wie folgt dargestellt. Um die Übersichtlichkeit zu erhöhen, werden die für diese Funktion nicht benötigten Rahmenbedingungen und Prozessbausteine aus den generischen Prozessbausteinen weggelassen. Es werden nur die jeweilige Kategorie bei den Rahmenbedingungen bzw. Prozessmodellen und der abstrakte Prozessbaustein bei den generischen Prozessmodellen belassen. Die zugewiesenen Prozessbausteine zu b) und c) werden ebenfalls ausgeblendet. Zu den Rahmenbedingungen „Unterlagen: Hypothekarkredit“ und „Kredithöhe: Hypothekarkredit“ würde jeweils eine Beziehung zu dem jeweiligen Prozessbaustein eingefügt (analog zu „Kreditrisiko: ja“ und „PB0.1 Kreditrisiko ermitteln“). Die Rahmenbedingungen „Unterlagen: Hypothekarkredit“ und „Kredithöhe: Hypothekarkredit“ wurden in einem Kasten zusammengefasst, um zu verdeutlichen, dass beide ausgewählt werden müssen, um den Musterprozess hinzuzufügen.*

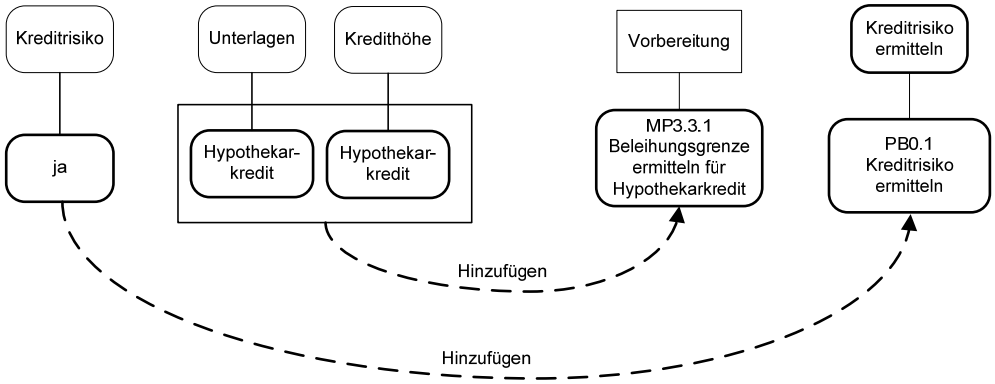



**Abbildung 102: Prozessanpassungsregeln**

*Da der Musterprozess durch die Prozessbausteine nur aus variablen Teilen besteht und keine zusätzlichen Prozessschritte enthält, könnte auch alternativ zu der Rahmenbedingung „Unterlagen: Hypothekarkredit“ als Prämisse (b) der Prozessbaustein PB3.3.1.2 vom Abhängigkeitstyp „Entfernen“ aus dem Musterprozess entfernt werden, wobei der Musterprozess vorher hinzugefügt werden muss. Würde (c) ausgewählt werden, könnte der Prozessbaustein PB3.3.1.1 durch den Abhängigkeitstyp „Entfernen“ aus dem Musterprozess MP3.3.1 entfernt werden.*

***Funktion „Einbauregel definieren“***

Mit Hilfe der Einbauregeln werden die Anordnungsbeziehungen zwischen den einzelnen Prozessbausteinen definiert. Dazu wird ein Prozessbaustein als Prämisse ausgewählt. Anschließend wird der Abhängigkeitstyp bestimmt. Mögliche Ausprägungen sind *„ist Vorgänger von“*, *„ist Nachfolger von“*, *„ist parallel zu“* oder *„ist zugehörig zu“*. Die Beziehungstypen sind umkehrbar, sie gelten daher auch in die andere Richtung. Im nächsten

Schritt werden ein oder mehrere Prozessbausteine als Konklusionen gewählt. Es wird automatisch *„für jeden Prozessbaustein, der als Konklusion gewählt wurde, … eine zweite Einbauregel mit entgegengesetzter Richtung“* erstellt. [Rupp02, 75ff]

***Beispiel 4.3.1.6***

*Als Prämisse werden jeweils die folgenden Prozessbausteine ausgewählt:*

*a) Auswahl „PB0.1 Kreditrisiko ermitteln“*
*b) Auswahl „PB3.1 Kredit beantragen“*
*c) Auswahl „PB3.2.1.1 Unterlagen beschaffen“*
*d) Auswahl „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“*

*Für alle wird der Abhängigkeitstyp „ist Vorgänger von“ festgelegt.*

*Für die Konklusionen werden folgende Prozessbausteine ausgewählt, wobei wiederum Prozessbausteine unter dem gleichen Punkt zugeordnet werden:*

*a) Auswahl „PB0.2 Kreditantrag ablehnen“ und „PB0.3 Kreditantrag bewilligen“*
*b) Auswahl „PB3.2.1.1 Unterlagen beschaffen“ und „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“*
*c) Auswahl „PB3.2.1.2 Kredithöhe anhand der Unterlagen für Privatkredit ermitteln“*
*d) Auswahl „PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“*

*Folgende Abbildung zeigt die Prozessbausteine mit den zugeordneten Einbauregeln grafisch:*

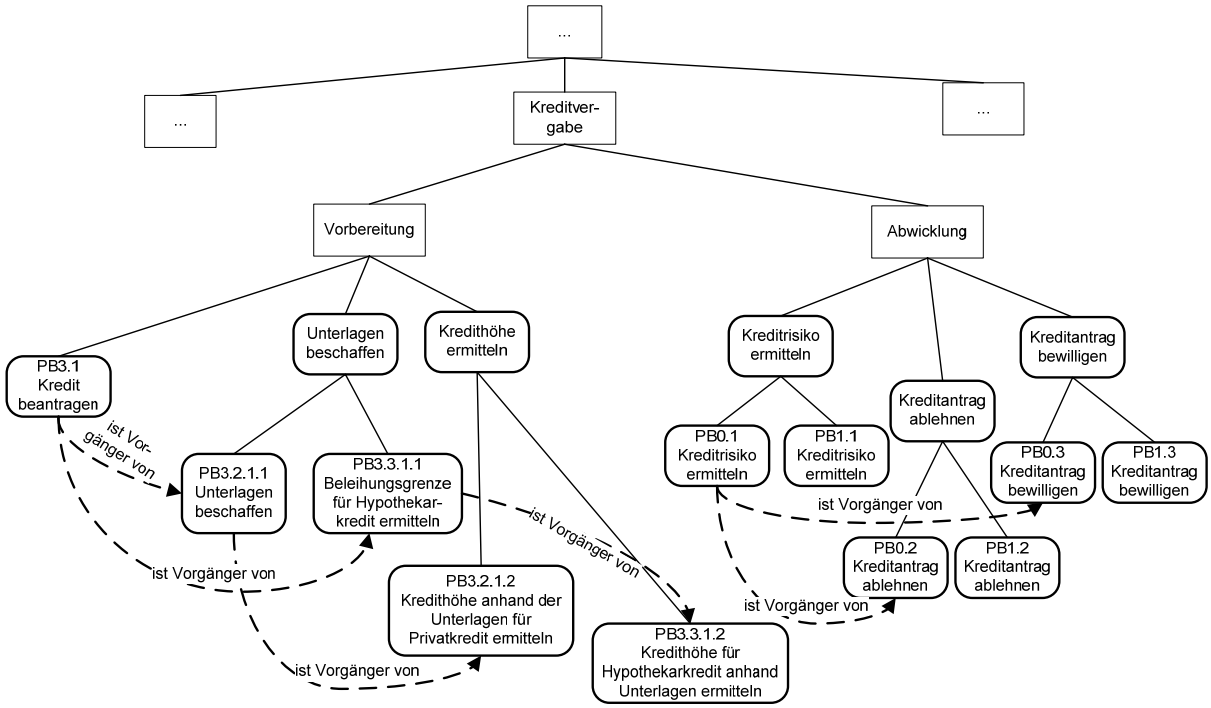



**Abbildung 103: Generische Prozessbausteine mit Einbauregeln**

*Automatisch werden folgende Abhängigkeitstypen eingefügt:*

a) *Für „‚PB0.1 Kreditrisiko ermitteln' **ist Vorgänger von** ‚PB0.2 Kreditantrag ablehnen' zusätzlich „‚PB0.2 Kreditantrag ablehnen' **ist Nachfolger von** ‚PB0.1 Kreditrisiko ermitteln'"*
   *Für „‚PB0.1 Kreditrisiko ermitteln' **ist Vorgänger von** ‚PB0.3 Kreditantrag bewilligen'" zusätzlich „‚PB0.3 Kreditantrag bewilligen' **ist Nachfolger von** ‚PB0.1 Kreditrisiko ermitteln'"*
b) *Für „‚PB3.1 Kredit beantragen' **ist Vorgänger von** ‚PB3.2.1.1 Unterlagen beschaffen'" zusätzlich „‚PB3.2.1.1 Unterlagen beschaffen' **ist Nachfolger von** ‚PB3.1 Kredit beantragen'"*
   *Für „‚PB3.1 Kredit beantragen' **ist Vorgänger von** ‚PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln'" zusätzlich „‚PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln' **ist Nachfolger von** ‚PB3.1 Kredit beantragen'"*
c) *Für „‚PB3.2.1.1 Unterlagen beschaffen' **ist Vorgänger von** ‚PB3.2.1.2 Kredithöhe anhand der Unterlagen für Privatkredit ermitteln'" zusätzlich „‚PB3.2.1.2 Kredithöhe anhand der Unterlagen für Privatkredit ermitteln' **ist Nachfolger von** ‚PB3.2.1.1 Unterlagen beschaffen'"*
d) *Für „‚PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln' **ist Vorgänger von** ‚PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln'" zusätzlich „‚PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln' **ist Nachfolger von** ‚PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln'"*

*Auf die grafische Darstellung dieser zusätzlichen Beziehungen wird verzichtet und der Interpretation des Lesers überlassen.*

### 4.3.2 Suche nach geeigneten Prozesslösungen

Mit der Erfassung der generischen Rahmenbedingungen, der generischen Prozessbausteine bzw. Musterprozesse und der Gestaltungsregeln wurde der Grundstein für die Wiederverwendung von Prozesswissen gelegt [Rupp02, 67ff]. Dazu wird das vorgestellte Vorgehensmodell aus Abbildung 88 (Seite 113) von Rupprecht näher erklärt. Die Angabe der Schritte bezieht sich auf die Schritte des Vorgehensmodells.

**Projektrahmenbedingungen setzen (Schritt 1):** Während der Projektinitialphase werden im ersten Schritt die Projektrahmenbedingungen für das gesamte Projekt gesetzt, was eine Auswahl der Einflussgrößen und der entsprechenden Ausprägung aus den generischen Rahmenbedingungen bedeutet. Diese Rahmenbedingungen werden *„dem obersten Knoten in der Prozesshierarchie"* zugeordnet. Dabei wird auf die Funktion „Projektrahmenbedingungen setzen" zurückgegriffen.

***Funktion „Projektrahmenbedingungen setzen"***

Zu Beginn werden eine oder mehrere Rahmenbedingungen gewählt, indem *„Ausprägungen unterschiedlicher Einflussgrößen in der Ordnerstruktur für generische Rahmenbedingungen"* ausgewählt werden. Die gewählten Rahmenbedingungen werden im nächsten Schritt auf Kontextgestaltungsregeln überprüft, die bei Vorhandensein weitere Rahmenbedingungen

vorschlagen, welche vom Benutzer angenommen bzw. abgelehnt werden können. [Rupp02, 114ff]

***Beispiel 4.3.2.1***

*Es werden die Ausprägungen „Hypothekarkredit“ der Einflussgröße „Kredit“ ausgewählt (aus Abbildung 101).*

*Für die Projektrahmenbedingung „Hypothekarkredit“ der Einflussgröße „Kredit“ gibt es Kontextgestaltungsregeln vom Aktionstyp „Hinzufügen“ zu den Rahmenbedingungen „Hypothekarkredit“ der Einflussgröße „Unterlagen“ und „Hypothekarkredit“ der Einflussgröße „Kredithöhe“ (vgl. Abbildung 101).*

*Folgende Abbildung zeigt die ausgewählten Ausprägungen mit ihren Einflussgrößen.*

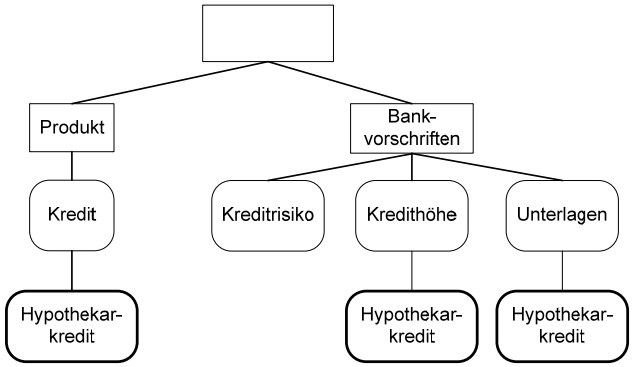



**Abbildung 104: "Spezifisches Modell von Projektrahmenbedingungen"[5]**

**Geeigneten Prozessfall suchen (Schritt 2):** In diesem Schritt werden die Projektrahmenbedingungen als Basis für die Suche nach einem *„Prozessfall aus früheren Projekten“* verwendet. Hier können CBR-Methoden (Case Based Reasoning) verwendet werden. Ein Prozessfall wird dann als geeignet erachtet, wenn die Projektrahmenbedingungen des früheren Projekts den Projektrahmenbedingungen des neuen Projekts gleich oder ähnlich sind. Ist ein geeigneter Prozessfall vorhanden, wird dieser im nächsten Schritt wieder verwendet (Schritt 3), ansonsten wird *„ein geeigneter Musterprozess als Referenzmodell“* gesucht (Schritt 5).

***Beispiel 4.3.2.2***

*Es ist kein Prozessfall mit den aus Schritt 1 ausgewählten Projektrahmenbedingungen vorhanden.*

**Wiederverwendung des Prozessfalls (Schritt 3):** Die Wiederverwendung erfolgt in diesem Schritt, indem der Prozessfall aus Schritt 2 kopiert und als Ausgangspunkt wiederverwendet wird.

[5] In Anlehnung an das Beispiel aus [Rupp02, 68]

**Anpassung der Projektrahmenbedingungen (Schritt 4):** Eine eventuelle Anpassung des Modells der *„Projektrahmenbedingungen des wiederverwendeten Prozessfalles an den neuen Kontext“* geschieht manuell.

**Geeigneten Musterprozess suchen (Schritt 5):** Die Suche erfolgt – analog zu Schritt 2 – ebenfalls auf Basis der Projektrahmenbedingungen, die im ersten Schritt gesetzt wurden.

***Beispiel 4.3.2.3***

*Der Musterprozess „MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit“ wird vorgeschlagen, da er mit den Ausprägungen „Unterlagen: Hypothekarkredit“ und „Kredithöhe: Hypothekarkredit“ verknüpft ist (vgl. Abbildung 102 auf Seite 126).*

**Wiederverwendung des Musterprozesses (Schritt 6):** Ist ein geeigneter bzw. sind mehrere geeignete Musterprozesse vorhanden, kann der Modellierer diese Musterprozesse als eine vorläufige Prozesshierarchie *„in den neuen Prozessfall“* einbauen. Dabei kann er *„bei jedem Prozessbaustein, der im Musterprozess enthalten ist“*, sequenziell entscheiden, ob der Prozessbaustein im Prozessfall eingefügt werden soll. Dazu wird die Funktion „Musterprozess einbauen“ verwendet. [Rupp02, 112f]

***Funktion „Musterprozess einbauen“***

Ein Musterprozess wird vom Benutzer ausgewählt und unter dem Teilprozess in der vorliegenden Prozesshierarchie eingefügt. Dabei wird für jeden Prozessbaustein abgefragt, der in dem Musterprozess vorkommt, ob dieser eingefügt werden soll.

***Beispiel 4.3.2.4***

*Der im vorigen Schritt vorgeschlagene Musterprozess „MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit“ wird ausgewählt. In diesem Musterprozess sind die Prozessbausteine „PB3.3.1.1 Beleihungsgrenze für Hypothekarkredit ermitteln“ und „PB3.3.1.2 Kredithöhe für Hypothekarkredit anhand Unterlagen ermitteln“ enthalten. Der Benutzer bestätigt den Einbau dieser Prozessbausteine sequenziell. Somit wird folgendes Prozessmodell auf der zweiten Ebene der Prozesshierarchie eingefügt:*

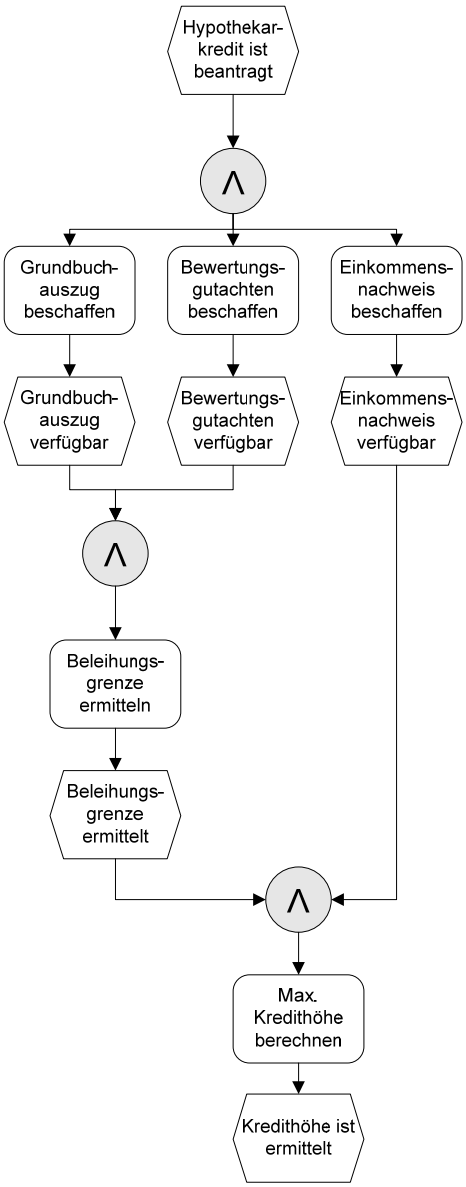



**Abbildung 105: Musterprozess**

**Weitere Rahmenbedingungen hinzufügen (Schritt 7):** Nach dem Einbau des Musterprozesses können Projektrahmenbedingungen ergänzt werden.

***Beispiel 4.3.2.5***

*Analog zu Schritt 1 werden nun zusätzlich benötigte Rahmenbedingungen gesetzt: Ausprägung „Ja“ der Einflussgröße „Kreditrisiko“. Folgende Abbildung zeigt nun das gesamte Modell des projektspezifischen Kontexts.*

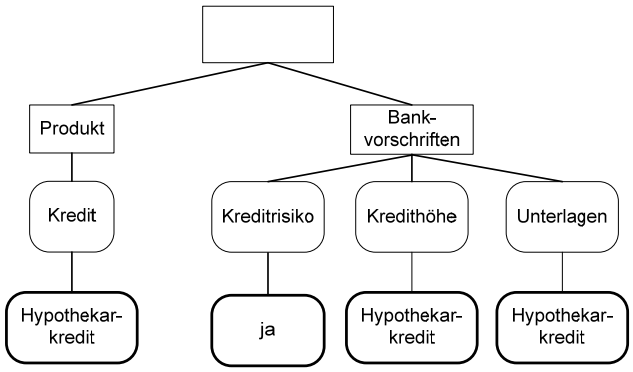



**Abbildung 106: Projektrahmenbedingungen**[6]

**Prozesshierarchie erstellen (Schritt 8):** Falls keine geeigneten Prozessfälle oder Musterprozesse gefunden werden, muss vom Benutzer eine Prozesshierarchie manuell erstellt werden, und zwar durch ein Prozessmodellierungswerkzeug. Hier kann auf Hilfsmittel wie z.B. ein Projektstrukturplan zurückgegriffen werden.

***Beispiel 4.3.2.6***

*Konnte in Beispiel 4.3.2.3 kein geeigneter Musterprozess gefunden werden, könnte der Benutzer Teilprozesse definieren, die in dem Projekt durchgeführt werden müssen. Angewendet auf das Rahmenbeispiel könnten dies folgende Teilprozesse sein: „Unterlagen", „Kredithöhe", „Kreditrisiko", die auf der gleichen Hierarchieebene angesiedelt sind. Diese Teilprozesse enthalten keine Prozessschritte.*

**Projektrahmenbedingungen setzen (Schritt 9):** Dem Teilprozess können auf den verschiedenen Ebenen weitere Projektrahmenbedingungen zugeordnet werden. Zu diesem Zweck wird die Funktion „Projektrahmenbedingungen setzen" erneut aufgerufen (vgl. Schritt 1 weiter oben). Nun kann wiederum nach einem Prozessfall oder einem Musterprozess gesucht werden (vgl. Schritt 2).

***Beispiel 4.3.2.7***

*Zu jedem Teilprozess aus Beispiel 4.3.2.6 werden in diesem Schritt Projektrahmenbedingungen zugewiesen.*

*a) Teilprozess „Unterlagen": Projektrahmenbedingung „Unterlagen: Hypothekarkredit"*
*b) Teilprozess „Kredithöhe": Projektrahmenbedingung „Kredithöhe: Hypothekarkredit"*
*c) Teilprozess „Kreditrisiko": Projektrahmenbedingung „Kreditrisiko: ja"*

*Zu diesen Projektrahmenbedingungen wird nun nach Prozessbausteinen bzw. Musterprozessen gesucht. Dieses Beispiel wird nicht mehr weiterverfolgt, sondern als Ausgangspunkt von Beispiel 4.3.2.6 verwendet.*

[6] In Anlehnung an das Beispiel aus [Rupp02, 68]

### 4.3.3 Anwendung von gefundenen Prozesslösungen

**Prozessmodellinstanzen anpassen (Schritt 10):** Die Prozessmodellinstanzen müssen mit Hilfe der Funktion „Teilprozess anpassen“ an die Projektrahmenbedingungen angepasst werden.

***Funktion „Teilprozess anpassen“***

Diese Funktion kann aus den Funktionen „Prozessbaustein einbauen“ bzw. „Musterprozess einbauen“ aufgerufen werden. Anschließend werden die Projektrahmenbedingungen des Teilprozesses ermittelt. Für diese Projektrahmenbedingungen werden eventuell vorhandene Prozessanpassungsregeln gesucht. Wird eine Prozessanpassungsregel gefunden, wird der Vorschlag generiert, diesen Prozessbaustein bzw. Musterprozess je nach Abhängigkeitstyp hinzuzufügen oder zu entfernen. Diese Vorschläge können nun vom Benutzer angenommen bzw. abgelehnt werden. Wird er nicht akzeptiert, *„wird die nächste Projektrahmenbedingung ausgewertet“*. Wird der Vorschlag akzeptiert, wird der Prozessbaustein bzw. Musterprozess je nach Abhängigkeitstyp eingebaut oder entfernt. Beim Aktionstyp „Hinzufügen“ werden zusätzlich eventuelle Prozessvarianten *„zur Auswahl angeboten“*. [Rupp02, 119ff]

Das Einbauen des Prozessbausteins erfolgt mit der Funktion „Prozessbaustein einbauen“ und des Musterprozesses mit der Funktion „Musterprozess einbauen“. Dabei wird untersucht, ob dieser bereits in der Prozesshierarchie enthalten ist. Der Modellierer kann jedoch trotz Konflikt den Prozessbaustein oder Musterprozess einbauen. Der Einbau kann dabei entweder aufgeklappt, d.h. *„direkt in die Prozess-Struktur des in der Prozessmodellinstanz selektierten“* Teilprozess hineinkopiert werden oder als Teilprozess, und somit *„eine Hierarchie-Ebene tiefer“*, eingefügt werden. Anschließend wird ausgewertet, ob weitere Einbauregeln enthalten sind (Funktion „Einbauregeln auswerten“). [Rupp02, 110]

In der Funktion „Einbauregeln auswerten“ wird untersucht, ob weitere Prozessbausteine mit diesem soeben ausgewählten Prozessbaustein durch die möglichen Abhängigkeitstypen verknüpft sind (*„ist Vorgänger von“*, *„ist Nachfolger von“*, *„ist parallel zu“* oder *„ist zugehörig zu“*). [Rupp02, 118]

Wenn durch eine Einbauregel der Vorschlag zum Hinzufügen oder Entfernen eines weiteren Prozessbausteins akzeptiert wird, wird dieser automatisch – sofern er vom Aktionstyp *„ist Nachfolger von“* oder *„ist Vorgänger von“* ist – vor oder nach dem Prozessbaustein eingefügt. [Rupp02, 110f]

***Beispiel 4.3.3.1***

*Ausgehend von dem in Beispiel 4.3.2.4 eingefügten Musterprozess „MP3.3.1 Beleihungsgrenze ermitteln für Hypothekarkredit“ und den in Beispiel 4.3.2.5 gesetzten Rahmenbedingungen wird nach Prozessanpassungsregeln zu der Rahmenbedingung „Kreditrisiko: Ja“ gesucht. Diese Projektrahmenbedingung: „Kreditrisiko: ja“ verfügt über eine Prozessanpassungsregel vom Abhängigkeitstyp „Hinzufügen“, und zwar den Prozessbaustein „PB0.1 Kreditrisiko ermitteln“ (vgl. Abbildung 102).*

*Die Prozessvariante (PB1.1) wird vorgeschlagen. Im Rahmen dieses Beispiels wird der Prozessbaustein PB0.1 ausgewählt und aufgeklappt auf der gleichen Hierarchieebene hinzugefügt.*

*Für den Prozessbaustein „PB0.1 Kreditrisiko ermitteln“ gibt es zwei Prozessbausteine, die dem Benutzer mit Hilfe der Einbauregel vorgeschlagen werden („PB0.2 Kreditantrag ablehnen“ und „PB0.3 Kreditantrag bewilligen“). Die vorgeschlagenen Prozessbausteine können sequenziell akzeptiert und ebenfalls auf der gleichen Hierarchieebene der Prozesshierarchie kopiert und eingefügt werden.*

*Mit Hilfe der Funktion „Einbauregel definieren“ wurden folgende Einbauregeln definiert:*

> *Für „‚PB0.1 Kreditrisiko ermitteln’* ***ist Vorgänger von*** *‚PB0.2 Kreditantrag ablehnen’ zusätzlich „‚PB0.2 Kreditantrag ablehnen’* ***ist Nachfolger von*** *‚PB0.1 Kreditrisiko ermitteln’“*
> *Für „‚PB0.1 Kreditrisiko ermitteln’* ***ist Vorgänger von*** *‚PB0.3 Kreditantrag bewilligen’“ zusätzlich „‚PB0.3 Kreditantrag bewilligen’* ***ist Nachfolger von*** *‚PB0.1 Kreditrisiko ermitteln’“*

*Aus diesem Grund werden diese Prozessbausteine (PB0.2 und PB0.3) nach PB0.1 eingebaut und es ergibt sich somit folgendes Prozessmodell.*

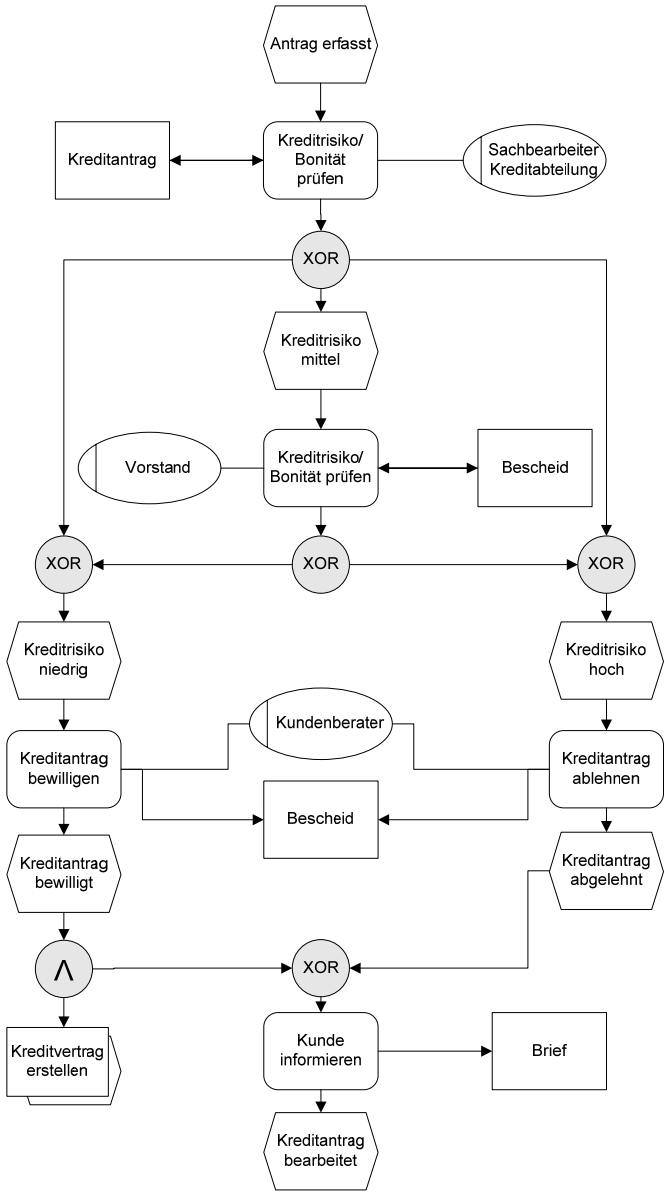



**Abbildung 107: Anhand der Einbauregeln zusammengefügte Prozessbausteine**

Dies ist *„ein erster Entwurf der Prozessmodellinstanzen“*, der in der Projektabwicklungsphase angepasst wird. [Rupp02, 125]

**Prozessmodellinstanzen manuell anpassen (Schritt 11):** Nun können die Prozessmodellinstanzen mit dem verwendeten Prozessmodellierungswerkzeug manuell angepasst werden.

***Beispiel 4.3.3.2***

*Die beiden einzelnen Prozessmodelle (vgl. Abbildung 105 und Abbildung 107) werden nun manuell angepasst und es wird die Funktion „Antrag ausfüllen“ eingefügt.*

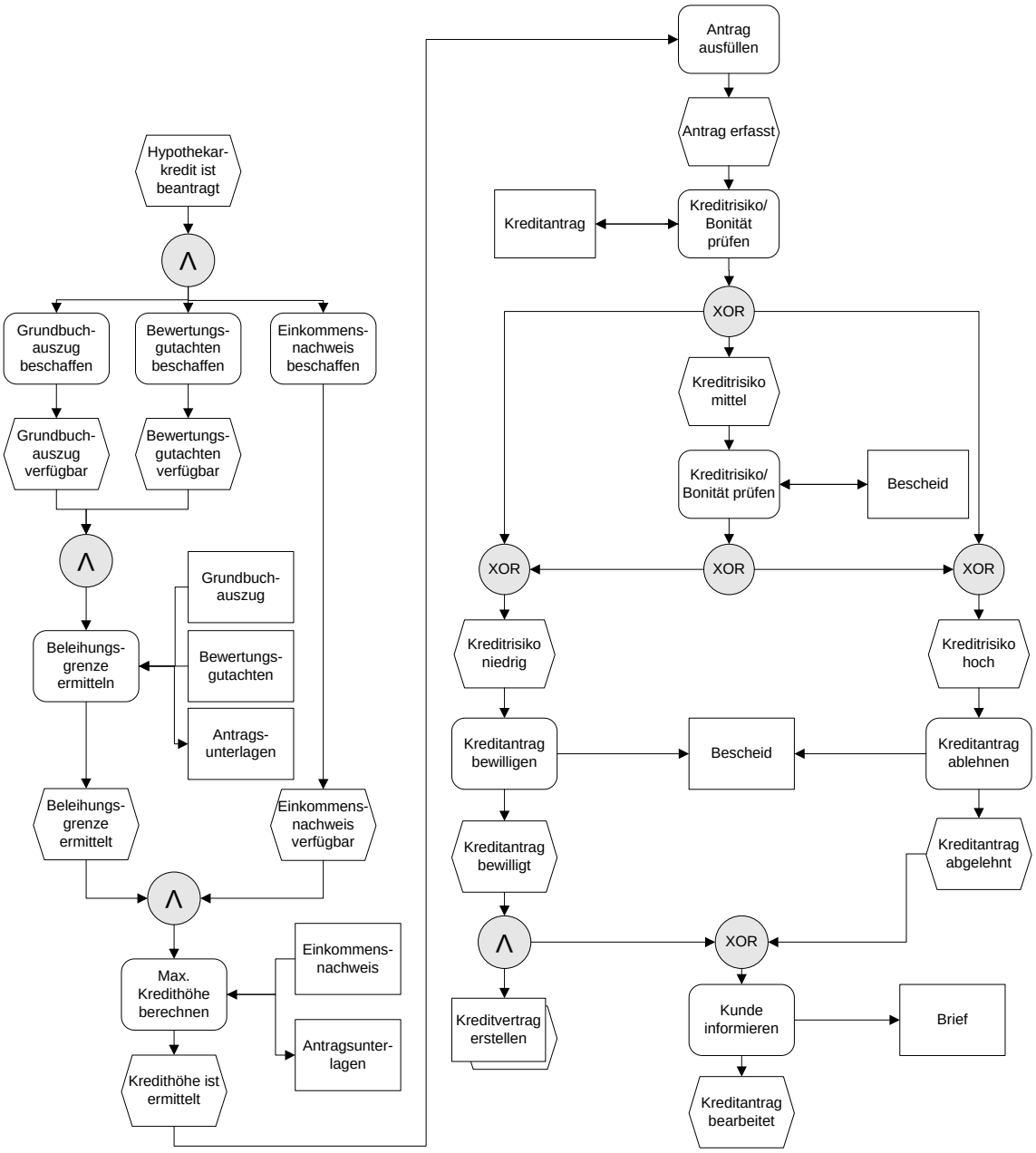



**Abbildung 108: Manuell angepasste Prozessmodellinstanz**

*Zur besseren Übersichtlichkeit können wieder durch Prozesswegweiser die Prozessschritte in einem eigenen Prozess nachgelagert werden. Somit kann dasselbe Prozessmodell wie in Abbildung 79 auf Seite 98 erstellt werden.*

**Manuelle Änderung und manuelles Hinzufügen von Projektrahmenbedingungen (Schritt 14):** Während der Laufzeit des Projekts können wiederum manuell Änderungen an den Projektrahmenbedingungen vorgenommen werden und neue definiert werden.

**Prozessmodellinstanzen anpassen (Schritt 15):** Wurden Änderungen durchgeführt, können in diesem Schritt eventuelle Inkonsistenzen zwischen Projektrahmenbedingungen und den Prozessmodellinstanzen durch die Funktion „Teilprozess anpassen“ beseitigt werden, indem wieder Vorschläge vom System generiert werden. Eine permanente Konsistenzprüfung zwischen diesen beiden Komponenten erfolgt nicht. In diesem Schritt wird genau wie in Schritt 10 vorgegangen. Nach diesem Schritt wird wieder Schritt 11 aufgerufen.

Die Schritte 11, 14 und 15 werden solange wiederholt bis keine Änderungen mehr an den Rahmenbedingungen und daher auch keine weiteren Anpassungen der Prozessmodellinstanzen durchzuführen sind.

### 4.3.4 Überprüfung der angepassten Prozesslösungen

Durch die Funktion „Teilprozess anpassen“ können *„Widersprüche zwischen Prozessmodellinstanzen und Projektrahmenbedingungen“* beseitigt werden. Eine automatische Konsistenzprüfung erfolgt nicht, sondern muss vom Benutzer angestoßen werden. Dies erfolgt im Schritt 15, der im vorigen Abschnitt erklärt wurde.

### 4.3.5 Integration neuer Prozesslösungen in die Wissensbasis

**Definition des neu geschaffenen Wissens (Schritt 12):** Sollten in Schritt 11 *„wiederverwendbare Strukturen erkennbar“* sein, kann die Wissensbasis entsprechend erweitert werden. Dazu werden wieder die Funktionen zur Definition neuer *„Rahmenbedingungen, Prozessbausteine, Musterprozesse und Gestaltungsregeln“* verwendet (vgl. Abschnitt 4.3.1). Dies bedeutet, dass die generischen *„Rahmenbedingungen, Prozessmodelle und Gestaltungsregeln“* um neue Strukturen durch den Benutzer erweitert werden.

**Pflege und Verwaltung der Wissensbasis (Schritt 13):** Für die *„Verwaltung und regelmäßige Pflege der Wissensbasis“* werden ebenfalls Funktionen spezifiziert. Die Funktionen umfassen unter anderem Funktionen für die *„Verwaltung und Suche generischer Prozessmodelle und generischer Rahmenbedingungen“* [Rupp02, 103ff].

**Erweiterung der Wissensbasis um den Prozessfall (Schritt 16):** Nachdem das Projekt abgeschlossen ist oder aber *„auch zu einem früheren Zeitpunkt“*, wird die Fallbasis um den Prozessfall erweitert.

# 5 Fallbasierter Ansatz nach Krampe

In Kapitel 2.2.2.4 wurde eine allgemeine Einführung in das fallbasierte Schließen gegeben. In diesem Abschnitt wird nun der fallbasierte Ansatz nach [Kram98] untersucht. Wie die meisten anderen fallbasierten Ansätze verwendet auch dieser nicht nur fallbasierte Methoden. Es handelt sich somit um einen hybriden Ansatz, der folgende Problemlösungsstrategien verwendet: [Kram98, 114]

- Modellbasiertes Schließen
- Schemabasiertes Schließen
- Fallbasiertes Schließen

Diese Strategien sind auf unterschiedlichen Abstraktionsniveaus angesiedelt. Das *Modellbasierte Schließen* ist das abstrakteste, wobei hier allgemeine Regeln formuliert werden, die *„auf viele Modelle“* angewendet werden können. Dazu hat Krampe Metamodellkonsistenzregeln in seinen Ansatz aufgenommen, die bereits weiter oben erklärt wurden. Sie gewährleisten die Konsistenz der Entwürfe bereits auf der Metamodellebene. In der nächsten Abstraktionsstufe ist das *Schemabasierte Schließen* angesiedelt, für das hier Anwendungsdomänekonsistenzregeln (Referenzmodellregeln) definiert wurden, um strukturelle Einheiten zu erhalten. Das *Fallbasierte Schließen* ist auf der untersten Abstraktionsebene angesiedelt. Die Fallbeispiele beschreiben *„die Situationen für konkrete Informationssysteme“*. [Kram98]

Es wurde beispielsweise in Bakhtari eine Kategorisierung der Aufgaben vorgenommen, die mehr oder weniger durch diesen Ansatz unterstützt werden: Routine Aufgaben, Innovative Aufgaben und Kreative Aufgaben [BaBS94]. *Routine Aufgaben* bezeichnen Aufgaben, die *„Schritt für Schritt von Anfang bis zum Ende gelöst werden“* können. Diese können von diesem Ansatz gut unterstützt werden, und zwar beispielsweise durch den integrierten modellbasierten Ansatz mit Hilfe der Metamodellkonsistenzregeln und durch den integrierten schemabasierten Ansatz mit Hilfe der Referenzmodellregeln. *Innovative Aufgaben* bezeichnen Aufgaben, die *„Schritt für Schritt bis zu einem Teilziel oder Entscheidungspunkt gelöst werden“* können. Es ist hier eine Koordination zwischen unterschiedlichen, voneinander abhängigen Tätigkeiten notwendig. Diese Art von Aufgaben kann *„durch Referenzmodelle und Fallbeispiele“* gelöst werden. Ein Beispiel wäre eine Prozesskette anzupassen, *„die von anderen Modellen abhängen kann und selbst weitere Modelle beeinflusst“*. *Kreative Aufgaben* können nicht standardisiert werden und somit auch nicht von diesem Werkzeug unterstützt werden. Ein Beispiel wäre ein neu aufgetretenes Problem für das in der Fallbibliothek keine Lösung enthalten ist. [Kram98, 115]

Bevor die Vorgehensweise der Wiederverwendung detailliert anhand des Rahmenbeispiels erklärt wird, werden zuerst die verwendeten Konzepte erläutert.

## *5.1 Verwendete Konzepte*

Dieser Ansatz verwendet Fallbeispiele und konzeptionelle Entwürfe, die in einer Fallbibliothek bzw. einer Entwurfsbibliothek (Wissensbasis) gespeichert werden [Kram98, 5ff] und auf die während der reaktiven Wiederverwendung zurückgegriffen wird.

*Fallbeispiele* beinhalten eine Suchanfrage (Query-Teil) an eine existierende Entwurfsbibliothek, eine Entwurfslösung (Solution-Teil) und einen Anwendungskontext, *„in dem die Entwurfslösung zu sehen ist“*. Die *Fallbibliothek* enthält Fallbeispiele. [Kram98, 76]

*Konzeptionelle Entwürfe* oder *Entwürfe* bestehen aus mehreren konzeptionellen Modellen, die logisch zusammengehören. Sie beschreiben *„die Gesamtheit der funktionellen Anforderungen, die an ein konkretes Informationssystem gestellt werden“* und bestehen aus verschiedenen Modellarten: Entity-Relationship-Modellen, Funktionsbäumen und ereignisgesteuerten Prozessketten. [Kram98, 75] In der *Entwurfsbibliothek* werden die konzeptionellen Entwürfe für Informationssysteme aus unterschiedlichen Anwendungsdomänen (z.B. einen konzeptionellen Entwurf einer Bank) gespeichert. [Kram98, 67ff]

Der *Anwendungskontext* besteht aus bestimmten Modellmerkmalen oder Komponentenmerkmalen, mit denen entschieden wird, ob ein Entwurf angenommen oder abgelehnt wird [Kram98, 75f]. Der Anwendungskontext kann auch als *„Wissen über spezifische Unternehmensstrukturen“* bezeichnet werden. Da Referenzmodelle abstrakt gehalten werden, verfügen sie über einen abstrakten Anwendungskontext, während Unternehmensmodelle, die spezielle Unternehmen darstellen, über einen spezifischeren Anwendungskontext verfügen. [Kram98, 35f]

Die *Lösung* eines Fallbeispiels ist im Solution-Teil festgehalten, der sich aus konzeptionellen Modellen und Modellkomponenten zusammensetzt [Kram98, 76]. *Konzeptionelle Modelle* oder *Modelle* formalisieren *„die funktionellen Anforderungen, die an ein Informationssystem gestellt werden“* [Kram98, 73]. Der Prototyp CBModeler, der diesen Ansatz verwendet, wurde mit Entity-Relationship-Modellen, Funktionsbäumen und ereignisgesteuerten Prozessketten getestet, obwohl auch andere Modelle verwendet werden können. Voraussetzung ist nur, dass die Modelle durch ein Metamodell beschrieben werden können und Metakomponenten, Beziehungen zwischen diesen Komponenten und Konsistenzregeln vorhanden sind [Kram98, 116]. Die *Modellkomponenten* oder auch *Komponente* sind die Elemente der verwendeten konzeptionellen Modelle. Die Elemente bei diesen Modellen sind Ereignis, Funktion, Entitätstyp und Attribut. [Kram98, 73f] Die Modelle und Komponenten müssen nach der Durchführung der Aufgabe nicht in den Solution-Teil übernommen werden [Kram98, 76].

Da die Fallbibliothek nicht alle für den Entwurf benötigten Entwurfselemente enthalten kann, integriert Krampe den Benutzer in den Prozess der Entwurfserstellung [Kram98, 52]. *Entwurfselemente* oder auch *Elemente* genannt sind Bausteine, die in den Entwürfen enthalten sind. Dabei kann ein Entwurfselement *„entweder eine Komponente (Funktion, Entitätstyp, Ereignis, Attribut) oder ein Modell (Entity-Relationship-Modell, Funktionsbaum, Prozesskette) sein“*. Die Entwurfselemente, die in dem neu erzeugten Fallbeispiel enthalten sein sollen und ausgewählt werden, sind *Entwurfsmerkmale*. [Kram98, 75]

Bei der ersten Anwendung dieses fallbasierten Ansatzes wird ein Referenzmodell importiert und mit jedem Durchlauf des CBR-Cycles wird die Wissensbasis um ein abgeleitetes Fallbeispiel erweitert. Während des Prozesses der reaktiven Wiederverwendung wird auf die eingangs erwähnten Wissensformen zurückgegriffen: *Methodenwissen*, *Domänenwissen*, *Fallwissen* und *Kontrollwissen*. Methodenwissen bezeichnet Wissen über die verwendete Modellierungsmethode und legt den Notationsrahmen fest. Domänenwissen bezeichnet das Wissen einer Anwendungsdomäne mitsamt den Strukturen, die das Unternehmen beschreiben.

Fallwissen ist das zu dem Methoden- und Domänenwissen ergänzende Wissen. Es beschreibt Wissen, das in einem spezifischen Anwendungskontext gültig ist. Für diese Kategorien werden auch Konsistenzkriterien definiert (Metamodellkonsistenz- und Referenzmodellregeln). [Kram98, 69] Darauf wird in der Reuse-Phase genauer eingegangen (Abschnitt 5.3.3).

Im Rahmen dieser Diplomarbeit wird der Fokus auf die zeitlich-logische Abfolge der Geschäftsprozesse (Prozessketten) gelegt und daher Entity-Relationship-Modelle und Funktionsbäume, die ebenfalls in den Fallbeispielen enthalten sind, außen vor gelassen, da sie nicht ausdrücklich Gegenstand dieser Diplomarbeit sind. Im zweiten Teil dieses Abschnittes wird das Vorgehensmodell nach Krampe erklärt und anhand des in Abschnitt 1.3 eingeführten Rahmenbeispiels auf seine Anwendbarkeit zur Wiederverwendung von Geschäftsprozessen untersucht. Abbildung 109 stellt den CBR-Cycle nach Krampe dar.

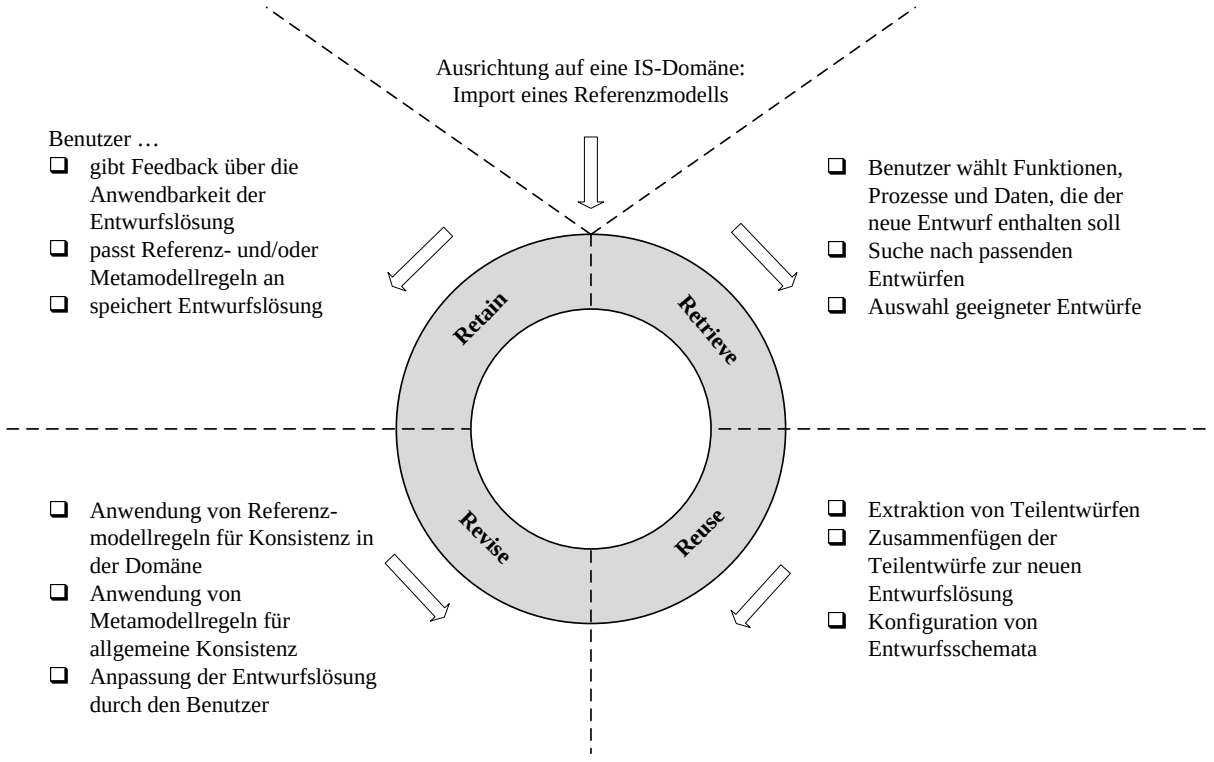



**Abbildung 109: CBR-Cycle nach Krampe [Kram98, 69]**

Für die Suche nach ähnlichen *Entwurfselementen* werden einfache Suchstrategien und Strategien der Ähnlichkeitsbestimmung verwendet, da es weniger wichtig ist in einem falladaptierenden System nach ähnlichen Elementen zu suchen, als sie im weiteren Verlauf anzupassen [Kram98, 116]. Die Vorgehensweise bei der Suche nach Entwurfselementen wird in Abschnitt 5.3.2 untersucht.

Dieser Ansatz unterstützt den Benutzer dabei *Teilentwürfe* zu identifizieren, *„die eine sinnvolle Einheit bilden“*. Dabei wird auf *„methoden- und domänenspezifisches Wissen“* zurückgegriffen. Durch die Verwendung kleinerer Teilentwürfe können Teile der vorhandenen Lösungen extrahiert und wiederverwendet werden. Um zu einer Lösung mit möglichst wenigen Überlappungen bzw. mit möglichst wenigen Fallbeispielen zu gelangen, werden die identifizierten Teilentwürfe iterativ in die Lösung aufgenommen. Dabei wird immer jener Teilentwurf zuerst verwendet, der die *maximale Überdeckung* liefert, also die meisten vom Benutzer ausgewählten Elemente enthält. Es können auch vorgefertigte

Standardelemente in die Lösung aufgenommen werden, die durch *Entwurfsschemata* dargestellt werden, und den variablen Teil der in der Wissensbasis vorhandenen Referenzmodelle ausmachen. Durch *Referenzmodelle* wird bereits eine Struktur der Anwendungsdomäne vorgegeben. [Kram98, 113f] Diese Vorgehensweise wird in Abschnitt 5.3.3 näher untersucht.

Die in der Reuse-Phase angepasste Lösung wird in der Revise-Phase konzeptintern durch Konsistenzregeln auf Metamodellkonsistenz und Anwendungsdomänenkonsistenz überprüft und kann anschließend extern durch den Benutzer überprüft werden, wobei notwendige Änderungen durchgeführt und die *„an das neue Informationssystem“* gestellten Anforderungen überprüft werden können [Kram98, 109ff]. Dies wird in Abschnitt 5.3.4 näher untersucht.

In der letzten Phase, der Retain-Phase kann der Benutzer Anpassungen bei den verwendeten Regeln (Metamodell- und/oder Referenzmodellregeln) vornehmen und die Anwendbarkeit des Fallbeispiels beurteilen [Kram98, 112f]. In Abschnitt 5.3.5 wird dies näher untersucht.

Um nach geeigneten Fällen suchen zu können, muss die Fallbibliothek entsprechend aufbereitet werden. [Kram98, 138ff] schlägt eine Darstellung eines Fallbeispiels mit folgenden Attributen vor:

| | |
|---|---|
| super(name) | Bezeichnung des vorherigen Knotens eines Fallbeispiels. Wird das Fallbeispiel von einem anderen Fallbeispiel abgeleitet, wird jener Name eingetragen, ansonsten wird der Wurzelknoten der Fallbibliothek eingetragen (case_library). |
| name | Bezeichnung des Fallbeispiels. Hier wird der Name des Unternehmens oder des Entwurfs angegeben, z.B. Entwurf einer Kreditvergabe |
| description | Kurzbeschreibung, z.B. Kreditvergabe mit Bonitätsprüfung mit den Risikoausprägungen hoch, mittel und niedrig. |
| context | Anwendungskontext, in dem der Fall anwendbar ist, beispielsweise zu erfüllende Rahmenbedingungen, z.B. Anwendung dieses Falles für Privatkunden. |
| outcome | Gelernte Erfahrungen während der Erstellung des Entwurfs. Diese werden in der vorliegenden Diplomarbeit vernachlässigt. Daher erfolgt hier keine Instanziierung. |
| alternate_ solution | Alternative Fälle, die ebenfalls angewendet werden können. |
| rejected_ solution | Fälle, die in Erwägung gezogen wurden, aber nicht angewendet werden können. |
| status | Ausprägung, inwieweit ein Fallbeispiel abgeschlossen ist. Mögliche Ausprägungen sind *learned* (gelernter Fall), *work-in-progress* (Fall noch in Bearbeitung) und *has-to-be-justified* (Anwendbarkeit wurde noch nicht bestätigt)[7]. |
| query | Entwurfselemente (Komponenten und Modelle) als Liste, die in der Retrieve-Phase erstellt wird. |
| er_models | Entity-Relationship-Modelle zu diesem Fallbeispiel in Form einer Liste. Diese werden in der vorliegenden Diplomarbeit vernachlässigt. |

[7] Die Anwendbarkeit wird in der Retain-Phase vom Modellierexperten bestimmt. Der Anwendungskontext in dem der Fall anwendbar ist, muss festgehalten werden [Kram98, 163f].

| | |
|---|---|
| | Daher erfolgt hier keine Instanziierung. |
| function_ trees | Liste der zu diesem Fallbeispiel gehörenden Funktionsbäume. Hier erfolgt ebenfalls keine Instanziierung, da dies in dieser Diplomarbeit vernachlässigt wird. |
| process_ chains | Liste der Prozessketten, die zu diesem Fallbeispiel gehören. |

**Tabelle 7: Elemente eines Fallbeispiels nach Krampe [Kram98, 138ff]**

In diesem Ansatz wurden Funktionsbäume, Entity-Relationship-Modelle und Ereignisgesteuerte Prozessketten verwendet, da sie bei der Modellierung von konzeptionellen Entwürfen sowohl in der Praxis als auch in der Forschung weit verbreitet sind. Dennoch ist dieser Ansatz methodenunabhängig, und es kann jede beliebige Methode verwendet werden, sofern ein Metamodell und entsprechende Konsistenzregeln vorhanden sind. [Kram98, 172]

Diese Phasen werden mit dem Prototypen CBModeler realisiert, welcher methoden- und domänenunabhängig ist. Es kann jede beliebige Modellierungsmethode eingesetzt werden, die durch ein *Metamodell* definiert ist und über *Metakomponenten*, *Beziehungen* untereinander sowie *Konsistenzregeln* verfügt. Somit kann auch jedes beliebige Referenzmodell verwendet werden, da die Modellierungsmethode angewendet wird und durch andere Entwurfsschemata sowie Referenzmodellregeln ersetzt werden. Die Kategorien anhand der Elemente unterschieden werden ändern sich, jedoch bleibt *„das Prinzip des Zugriffs auf einzelne Elemente durch Kategorien“* gleich. [Kram98, 116f] Die *Metamodellkonsistenz-* und *Referenzmodellregeln* können ebenfalls zur Identifikation von Teilentwürfen (vgl. Abschnitt 5.3.3) und zur Sicherung der Konsistenz (vgl. Abschnitt 5.3.4) verwendet werden.

## *5.2 Organisation der Wissensbasis*

Die Wissensdarstellung in der Entwurfsbibliothek erfolgt durch Bäume und gerichtete Graphen, einer speziellen Form von semantischen Netzen, die aus Knoten und Kanten bestehen. Krampe unterscheidet *Komponentenbäume*, *Modellbäume* und *Indexbäume*. [Kram98, 83f]

In den *Komponentenbäumen* werden *Komponenten* (Modellkomponenten) in einer Vererbungshierarchie dargestellt. Komponenten sind Entitätstypen, Funktionen, Ereignisse und Attribute und werden in den konzeptionellen Modellen verwendet. Eine Komponente kann laut Krampe entweder an die neue Problemstellung angepasst werden (*is-modified* Beziehung) oder analog zur Objektorientierung von einer anderen Komponente mit einer *is-a* Beziehung erben (Spezialisierung). Die erwähnten Beziehungen können nur zwischen Komponenten des gleichen Typs bestehen. Üblicherweise existiert eine große Anzahl an Komponentenbäumen, die meistens sehr klein sind, da sie unterschiedliche Varianten zusammenfassen und mit Hilfe der Spezialisierung (is-a) und Modifikation (is-modified) verbunden sind. [Kram98, 85f]

Folgende Abbildung zeigt die Struktur von Komponentenklassen, in denen die einzelnen Komponenten zusammengefasst sind [Kram98, 142].

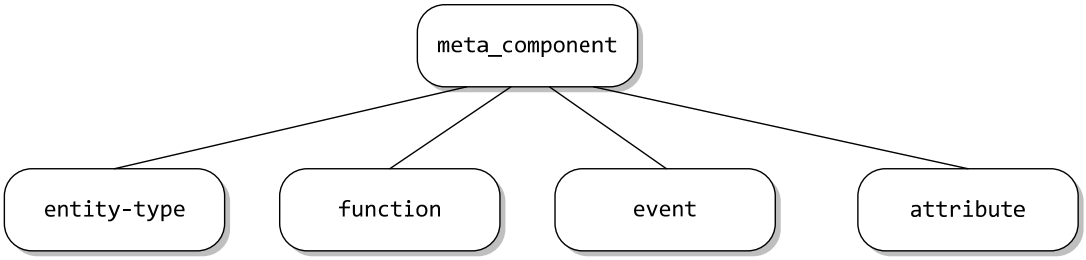



**Abbildung 110: Struktur von Komponentenklassen**

Abbildung 111 zeigt einen Ausschnitt eines Komponentenbaums. Die reaktive Wiederverwendung kann auch über Anwendungsdomänen hinweg angewandt werden. Der Entitätstyp Lieferant ist in einem Entwurf enthalten, der für ein Verkaufssystem eines Elektrogroßhandels erstellt wurde. Wird beispielsweise für ein Verkaufssystem für einen Buchhändler ein speziellerer Entitätstyp für Lieferanten benötigt, werden die zusätzlich benötigten Eigenschaften für diesen Entwurf mit einer Spezialisierungsbeziehung angefügt (is-a). [Kram98, 84]

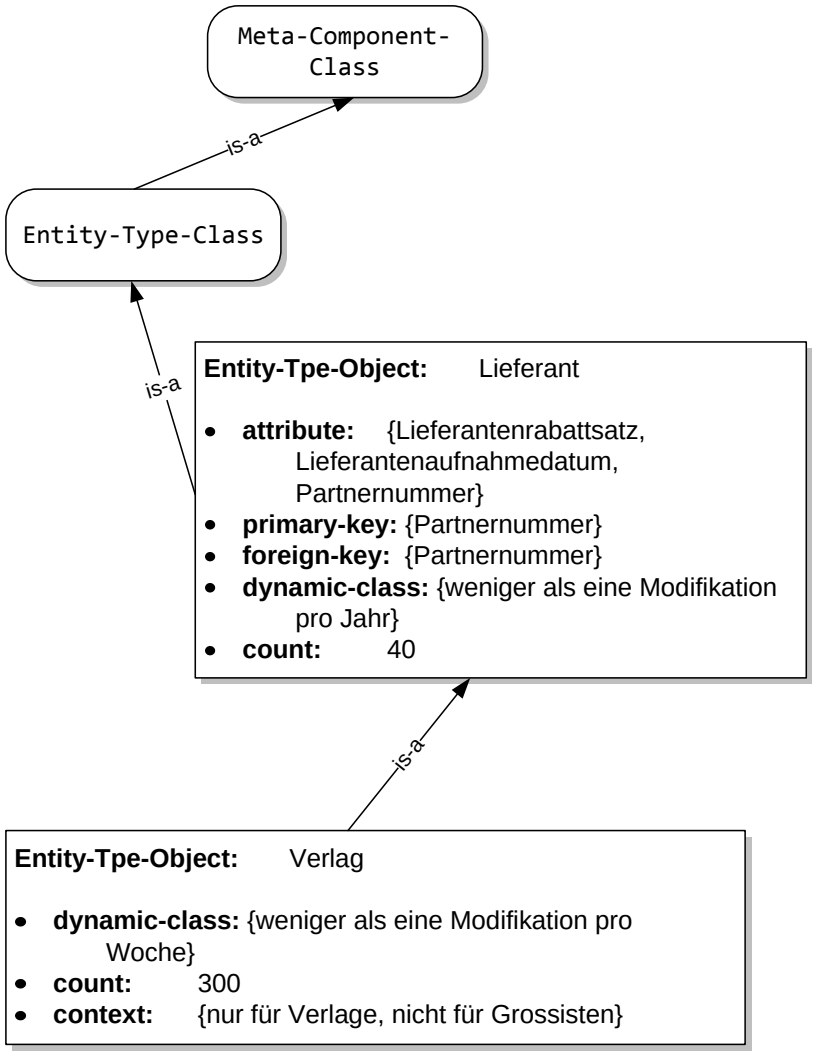



**Abbildung 111: Beispiel eines Komponentenbaums (Ausschnitt)**

Analog zu den Komponentenbäumen fassen die *Modellbäume* die konzeptionellen Modelle zu einer Vererbungshierarchie zusammen. Auch hier können Modelle durch Spezialisierung verfeinert oder angepasst werden (Modifizierung) und mit denselben Beziehungen wie die Komponentenbäume verbunden werden. Krampe definiert die Modelle Entity-Relationship-

Modelle, die die Datenstruktur mit ihren Attributen abbilden (Datensicht), Funktionsbäume, die die vom Informationssystem möglichen Funktionen abbilden (Funktionssicht) und ereignisgesteuerte Prozessketten (EPK), die die zeitlich-logische Abfolge von Geschäftsprozessen (Prozess- oder Steuerungssicht) darstellen. Die Entity-Relationship-Modelle und die Prozessketten werden durch gerichtete Graphen und die Funktionsbäume durch Bäume dargestellt. [Kram98, 87]

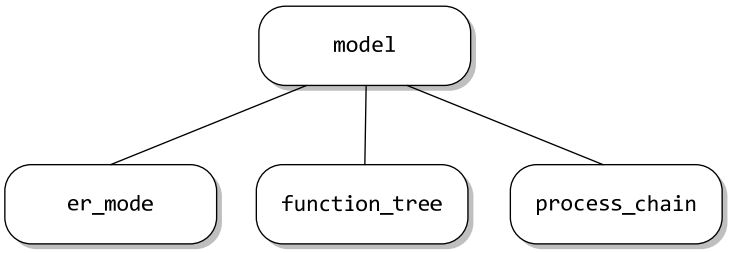



**Abbildung 112: Struktur von Modellklassen**

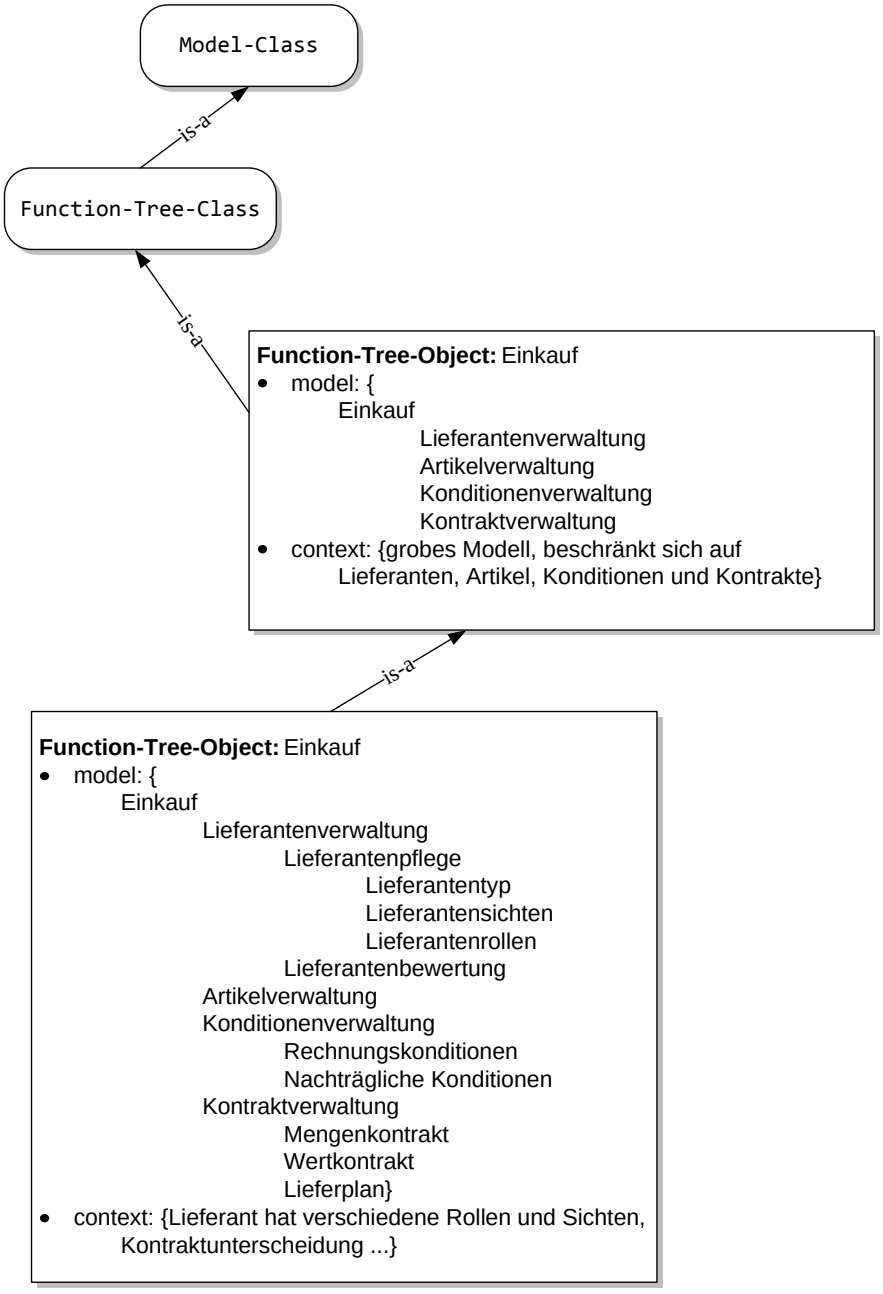



**Abbildung 113: Modellbaum (Ausschnitt) [Kram98, 86]**

Die einzelnen Entwurfselemente können während des Entwurfsprozesses durch Spezialisierung und Modifikation wieder verwendet werden. Es wird dabei die Vererbung der Entwurfsmerkmale ausgenutzt. [Kram98, 113]

Mit Hilfe von *Indexbäumen* kann eine Kategorisierung von Modellen und Komponenten vorgenommen werden [Kram98, 89ff]. Dabei werden konzeptionelle Modelle sowie Komponenten nach unterschiedlichen Unterscheidungsmerkmalen zusammengefasst, wobei zwischen allgemeinen Unterscheidungsmerkmalen und speziellen Unterscheidungsmerkmalen differenziert wird [KrLu96]. Allgemein wird zwischen Kriterien nach Funktionsbereich und Abstraktionsgrad unterschieden. Bei den speziellen Unterscheidungsmerkmalen unterteilt er die Kriterien in Sicht, Datenart und Datenzugriffsart. [Kram98, 78f]

*Funktionsbereiche* werden durch die in der Wissensbasis vorhandenen Referenzmodelle geprägt [Kram98, 78f]. Beispiele für Funktionsbereiche eines Handels-Referenzmodells finden sich in [BeSc96]. Beim *Abstraktionsgrad* wird zwischen dem abstrakten Referenzmodell und einem speziellen Unternehmensmodell unterschieden [Kram98, 78f].

Bei den *Sichten* wird zwischen Datensicht (Entitätstypen und Entity-Relationship-Modellen), Funktionssicht (Funktionsbäume und Funktionen) und Prozesssicht (Prozessketten, Ereignisse und Funktionen) unterschieden. *Datenarten* können in Stammdaten und Bewegungsdaten unterschieden werden. Stammdaten sind betriebswirtschaftliche und technische Daten. Bewegungsdaten sind Vormerkdaten, Transferdaten und Archivdaten. Bei der *Datenzugriffsart* wird unterschieden, was mit den Daten geschehen soll (z.B. verwalten, lesen, erzeugen, löschen, modifizieren, transformieren, importieren). [Kram98, 80ff]

Diese Einteilung in Unterscheidungsmerkmale dient der Vereinfachung der Suche innerhalb der Indexbäume. Zur Suche können auch verschiedene Merkmale kombiniert werden. [Kram98, 82]

## *5.3 Wiederverwendung anhand des Rahmenbeispiels*

Die meisten fallbasierten Ansätze gehen davon aus, dass ähnliche Fälle über eine ähnliche Problembeschreibung gefunden werden können [Kram98, 91]. Wess definiert das Dilemma dieser Vorgehensweise zur Auswahl der Fallbeispiele nach dieser Vorgehensweise folgendermaßen [Wess96]:

*"Das a posteriori Kriterium der Nützlichkeit von Problemlösungen wird daher, mangels anderer Alternativen, im fallbasierten Schließen auf den Begriff der Ähnlichkeit von Problemstellungen reduziert. Diese Vorgehensweise liegt in der Hoffnung begründet, daß in vielen Anwendungen die Ähnlichkeit von Problemstellungen die Brauchbarkeit für die Problemlösung impliziert. Diese oft implizite Hoffnung beim Einsatz fallbasierter Systeme spiegelt sich in der Literatur unter anderem in der für fallbasierte Anwendungen geforderten bereichsweisen Stetigkeit der betrachteten Domäne wider. Kleine Änderungen in der Problemstellung sollen nur zu kleinen Änderungen in den zu betrachtenden Lösungen führen."*

Für die Suche nach ähnlichen Fällen werden bei diesem fallbasierten Ansatz nicht die Problembeschreibungen verglichen, sondern der Benutzer hat die Möglichkeit, diejenigen

Entwurfselemente auszuwählen (Entwurfsmerkmale), die für die Erstellung der neuen Lösung nützlich sind. [Kram98, 91f] begründet dies wie folgt:

- Es kann nicht davon ausgegangen werden, dass kleine Änderungen in der Problemstellung nur kleine Änderungen in der Lösung nach sich ziehen. Jedes Problem ist einzigartig. Dies bedeutet, dass die Lösung trotz vieler Gemeinsamkeiten auch entscheidende Unterschiedlichkeiten aufweisen kann und somit in entscheidenden Mehraufwand bei der Lösung des Problems resultieren kann.
- Es kann dann zwischen Problem und Lösung unterschieden werden, *„wenn sich ein Entwurfsproblem so erschöpfend und umfassend beschreiben ließe“*, dass alle Informationen, die für die Problemlösung notwendig sind, beschafft werden könnten. Dies ist jedoch nie der Fall, da sich Anforderungen häufig ändern.
- Durch konzeptionelle Entwürfe wird die Problembeschreibung bereits formalisiert und die Implementierung erleichtert. Eine zusätzliche Formalisierung der Problembeschreibung ist daher überflüssig.

Das Vorgehensmodell umfasst in Anlehnung an [AaPl94] die vier Phasen Retrieve, Reuse, Revise und Retain. Damit beginnt die eigentliche Entwurfsarbeit [Kram98, 89ff]. Nachfolgende Tabelle zeigt eine Zuordnung der Phasen und der in Abschnitt 2.2.2.1 eingeführten Einteilung.

| **Einteilung** | **Krampe** |
|---|---|
| Integration von externen Prozesslösungen | Referenzmodellimport |
| Suche nach geeigneten Prozesslösungen | Retrieve |
| Anwendung von gefundenen Prozesslösungen | Reuse |
| Überprüfung der angepassten Prozesslösungen | Revise |
| Integration neuer Prozesslösungen in die Wissensbasis | Retain |

**Tabelle 8: Zuordnung der gemeinsamen Konzepte und Aufgaben nach Krampe**

### 5.3.1 Integration von externen Prozesslösungen

Wird dieser Ansatz das erste Mal verwendet, wird ein Referenzmodell in die Wissensbasis importiert und *„auf eine Anwendungsdomäne ausgerichtet“* [Kram98, 150f]. Dies ist notwendig, um auch bei der ersten Anwendung des CBR-Cycles auf vorhandenes Wissen zurückgreifen zu können. Durch den Import werden die Komponenten-, Modell- und Indexbäume erstellt und Konsistenzregeln (Metamodell- und Referenzmodellregeln) festgelegt, die in der Revise-Phase benötigt werden (vgl. Abschnitt 5.3.4). In diesem Abschnitt werden jedoch die Prozessmodelle aus Abschnitt 1.4 als Fallbeispiele nach dem Schema aus Tabelle 7 auf Seite 142 dargestellt.

Auf eine Erstellung der Wissensbasis, und somit einer Zuordnung der einzelnen Komponenten der ereignisgesteuerten Prozessketten zu Komponentenbäumen, der Prozessmodelle zu Modellbäumen und einer Kategorisierung in Indexbäumen wird verzichtet, da es die weitere Untersuchung nicht beeinflusst.

***Beispiel 5.3.1.1***

*Die Darstellung der Fallbeispiele, die in der Entwurfs- bzw. Fallbibliothek enthalten sind, ist in Abbildung 114 ('Kreditvergabe mit Bonitätsprüfung, Risiko hoch/mittel/niedrig'), Abbildung 115 ('Kreditvergabe mit Bonitätsprüfung, Risiko niedrig/hoch') und Abbildung 116 ('Privat- und Hypothekarkreditvergabe, Risiko hoch/mittel/niedrig') ersichtlich.*

```
case_1 :: {
  super(case_library) &
  name('Bank XYZ') &
  description('Kreditvergabe mit Bonitätsprüfung, Risiko hoch/mittel/niedrig') &
  context('Privatkunden') &
  outcome('') &
  alternate_solution('case_2') &
  rejected_solution('case_3') &
  status(learned) &
  query([model_1, model_2, object_1, object_56, object_103) &
  er_models(…) &
  function_trees(…) &
  process_chains([model_100]) }
```

**Abbildung 114: 'Kreditvergabe mit Bonitätsprüfung, Risiko hoch/mittel/niedrig'**

```
case_2 :: {
  super(case_library) &
  name('Bank YZ') &
  description('Kreditvergabe mit Bonitätsprüfung mit Risiko niedrig/hoch') &
  context('Privatkunden') &
  outcome('') &
  alternate_solution('case_1') &
  rejected_solution('case_3') &
  status(learned) &
  query([model_32, object_1, object_56, object_103) &
  er_models(…) &
  function_trees(…) &
  process_chains([model_101]) }
```

**Abbildung 115: 'Kreditvergabe mit Bonitätsprüfung, Risiko niedrig/hoch'**

```
case_3 :: {
  super(case_library) &
  name('Bank XY') &
  description('Privat- und Hypothekarkreditvergabe, Risiko hoch/mittel/niedrig') &
  context('Privatkunden, Kredit- und Hypothekantragsprüfung') &
  outcome('') &
  alternate_solution('case_57') &
  rejected_solution('case_63') &
  status(learned) &
  query([model_12, object_1, object_56, object_103]) &
  er_models(…) &
  function_trees(…) &
  process_chains([model_102, model_1021, model_1022]) }
```

**Abbildung 116: 'Privat- und Hypothekarkreditvergabe, Risiko hoch/mittel/niedrig'**

*Das Attribut super(case_library) bedeutet, dass die abgebildeten Fallbeispiele mit der case_library selbst verbunden und bedeutet somit, dass es sich hier um keine Spezialisierungen anderer Fallbeispiele handelt.*

*Der Name des Fallbeispiels ist in diesem Fall der einer Bank für die der Entwurf erstellt wurde. Unter alternate_solution wird auf andere Fälle verwiesen, beispielsweise wird bei „case_2“ auf „case_1“ verwiesen. Zu rejected_solution wird auf Fallbeispiele verwiesen, die in Erwägung gezogen, beispielsweise wurde für „case_2“ auch „case_3“ in Erwägung gezogen, jedoch abgelehnt wurden.*

*Auf das Vergeben von Werten von outcome, function_trees und er_models wird in dem Beispiel verzichtet. Die nicht instanziierten Elemente werden in diesen Abbildungen kursiv dargestellt.*

*Der Query-Teil ist bei diesen Fallbeispielen unerheblich, da diese vor ihrer Erstellung unter existierenden Entwurfselemente gesucht werden musste. Der Solution-Teil ist die eigentliche Lösung aufgeteilt auf die Attribute er_models, function_trees und process_chains als Liste. Unter dem Attribut process_chains werden die Prozessmodelle aus Abschnitt 1.4 angeführt. Die restlichen Attribute der Solution werden nicht beachtet.*

### 5.3.2 Suche nach geeigneten Prozesslösungen

In der Retrieve-Phase werden geeignete Fallbeispiele in der Fallbibliothek ausgewählt, die das Entwurfsproblem möglichst umfassend lösen sollen. Anfangs wird das neue Entwurfsproblem mit Hilfe der Entwurfsmerkmale beschrieben, indem Entwurfselemente ausgewählt werden. Das Ergebnis der Retrieve-Phase ist die Auswahl der Entwürfe, die für das neue Entwurfsproblem am besten passen. Krampe weist dieser Phase folgende drei Aufgaben zu: *Entwurfsmerkmale auswählen,* indem Elemente aus der Fallbasis ausgewählt werden, *nach passenden Entwürfen suchen* und einen oder mehrere *Entwurfskandidaten* aus der Entwurfsbibliothek *auswählen*. [Kram98, 89f]

Krampe führt dazu die Begriffe *Query-Case* und *Source-Case* ein. Ein Query-Case beinhaltet eine Suchanfrage (*Query*), die die Entwurfsmerkmale des neuen Entwurfs enthält. Dies wird im Query-Teil des Query-Cases abgelegt. Im Solution-Teil wird nach Ende des gesamten Prozesses der reaktiven Wiederverwendung (nach der Retain-Phase) die fertige Lösung (Entwurf) abgelegt und stellt ein neues Fallbeispiel in der Fallbibliothek dar. Ein Source-Case ist ein Fallbeispiel, das nach Anwendung dieser Phase ausgewählt wird, da er die ausgewählten Entwurfsmerkmale enthält. [Kram98, 89f]

Laut Krampe ist es sinnvoll, alle gefundenen Source-Cases in die Wiederverwendung einzubeziehen und aus diesen Teilentwürfe zu identifizieren, um die optimale Entwurfslösung zu erreichen [Kram98, 100] und durch geeignete Strategien eine Lösung zu generieren (vgl. Anwendung von gefundenen Prozesslösungen in Abschnitt 5.3.3).

In dieser Phase besteht der Input aus allen Elementen, die in den Modell- und Komponentenbäumen enthalten sind. Der gelieferte Output besteht aus einem neu erzeugten Fallbeispiel $c_{query}$, das in der *Query* die ausgewählten Entwurfselemente und einen leeren Solution-Teil enthält, aus den entsprechenden aktivierten Knoten der Modell- und Komponentenbäume sowie der lokalen Ähnlichkeit zwischen den aktivierten und

ausgewählten Knoten und aus aktivierten Fallbeispielen $c_{source}$ sowie der globalen Ähnlichkeit zwischen dem aktivierten Fallbeispiel und der Gesamtheit der Entwurfsmerkmale. [Kram98, 89ff] In Tabelle 9 sind der Input, der Output und die Datenstruktur des Algorithmus aus Abbildung 117 übersichtlich dargestellt. Die einzelnen Schritte des Algorithmus werden nachfolgend genau erklärt.

| | |
|---|---|
| ***Input*** | Modell- und Komponentenbäume |
| ***Output*** | $c_{query}$ |
| | $c_{source}$ |
| | lokale Ähnlichkeit zwischen $k_i$ und $k$ |
| | globale Ähnlichkeit zwischen den aktivierten Fallbeispielen *csource* und dem neuen Fallbeispiel $c_{query}$ |
| ***Elemente und Datenstrukturen*** | |
| $k_i$ … | Entwurfselement an der i-ten Stelle, kann ein Modell (z.B. EPK, Funktionsbaum oder Entity-Relationship-Modell) oder eine Komponente daraus sein (z.B. Funktion oder Ereignis) |
| *i…* | Index der Entwurfselemente, 1…m |
| *Query…* | Vom Benutzer ausgewählte Entwurfselemente ($k_1,\ldots,k_m$) |
| $c_{query}$ … | Neues Fallbeispiel, bestehend aus *Query* und leerer *Solution* |
| $c_{source}$ … | gesamte Menge der Source-Cases, die mindestens ein Entwurfselement enthalten, Source-Cases bestehen ebenfalls aus *Query* und *Solution* |
| *Solution…* | Lösung besteht aus Modellen (Entity-Relationship-Modellen, Funktionsbäumen und ereignisgesteuerten Prozessketten) |
| *lokale Ähnlichkeit* | Wird mit der Hilfsfunktion $sim_l(k_i,k)$ berechnet. Wertebereich zwischen 0 und 1 |
| *globale Ähnlichkeit* | Wird mit der Hilfsfunktion $sim(c_{query},c)$ berechnet. Wertebereich zwischen 0 und unendlich |
| ***Hilfsfunktion*** | |
| $sim_l(k_i,k)$ | Liefert als Rückgabewert ein Ähnlichkeitsmaß zwischen dem ausgewählten Entwurfselement $k_i$ und $k$. $k$ bezeichnet einen aktivierten Knoten des Baumes, wobei $k_i$ und $k$ derselben Indexkategorie zugeordnet sind, die verschiedene Spezialisierungen beinhaltet. Dabei wird der Abstand *„zum speziellsten, gemeinsamen Vorgänger“* berechnet.[8] |
| $sim(c_{query},c)$ | Liefert als Rückgabewert die Summe der lokalen Ähnlichkeiten zwischen den Entwurfselementen $k_i$ aus $c_{query}$ und $c$. $c$ bezeichnet einen aktivierten Source-Case aus der Menge csource, welche die gesamte Menge der Source-Cases enthält.[9] |

**Tabelle 9: Algorithmenbeschreibung zu Abbildung 117[10]**

[8] Genaue Darstellung der Hilfsfunktion vgl. [Kram98, 98f].
[9] Genaue Darstellung der Hilfsfunktion vgl. [Kram98, 98ff].
[10] Algorithmenbeschreibung samt den Datenstrukturen wurde aus [Kram98, 89ff] entnommen.

Die oben genannten Aufgaben der Retrieve-Phase werden wie folgt algorithmisch dargestellt:

Schritt 1: Wähle benötigte Elemente *(Query = {$k_1$,…,$k_m$})*
Schritt 2: Erzeuge neues Fallbeispiel (Query-Case): $c_{query}$ = (*Query, Solution = ϕ)*
Schritt 3: Für jedes Element $k_i \in c_{query}$*:*
Aktiviere Knoten des Modell- oder Komponentenbaums von $k_i$
Berechne lokale Ähnlichkeit zwischen aktivierten Knoten und $k_i$
Aktiviere alle Fallbeispiele, die aktivierte Knoten enthalten
Schritt 4: Berechne globale Ähnlichkeit zwischen aktivierten Fallbeispielen und $c_{query}$

**Abbildung 117: Ablauf der Retrieve-Phase nach Krampe [Kram98, 90]**

**Elementauswahl (Schritt 1):** Um das neue Entwurfsproblem zu beschreiben, wählt der Benutzer die benötigten Elemente aus, die im Query-Teil des *Query-Cases* abgelegt werden. Die ausgewählten Elemente sind wie bereits erwähnt Entwurfsmerkmale, über die das neue Entwurfsproblem verfügen soll. Der Solution-Teil des Query-Cases ist zu Beginn noch leer und enthält nach dem Durchführen aller Teilaufgaben die endgültige Lösung. [Kram98, 89f]

Für die Elementauswahl schlägt Krampe entweder eine Auswahl von Elementtypen (Entity-Relationship-Modell, Funktionsbaum, Prozesskette, Ereignis, Funktion, Entitätstyp, Attribut) oder einer Indexkategorie (Abstraktionsbereich, Funktionsbereich, Datenart, Sicht, Datenzugriffsart) oder eine Kombination aus beiden vor. Der Benutzer wählt also entweder Modelle oder Komponenten von Modellen oder beides aus, die exakt oder ungefähr der Problemstellung entsprechen. Wählt der Benutzer beispielsweise den Elementtyp „Funktionen“, werden alle Funktionen angezeigt, die in der Entwurfsbibliothek gespeichert sind. Abhängig von der Problemstellung wählt der Benutzer jene Funktionen aus, die er für seinen konzeptionellen Entwurf benötigt. Die ausgewählten Elemente werden als $k_i$ in der *Query* gespeichert. [Kram98, 78ff] Die grafische Darstellung des CBModeler-Retrievers ist in Krampe ersichtlich [Kram98, 152] und ist eine Möglichkeit die Elementauswahl zu gestalten.

***Beispiel 5.3.2.1***

*Gemäß der Aufgabenstellung in Abschnitt 1.3.1 werden jene Funktionen aus Abschnitt 1.4 ausgewählt, die für die Erstellung des Geschäftsprozesses „Hypothekantrag einer Bank bearbeiten“ notwendig sind und anschließend in einer Query gespeichert. Für diesen Geschäftsprozess wird folgende Query erzeugt:*

*Query = { 'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren', 'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln' }*

*Diese Query stellt nun alle vom Benutzer ausgewählten Entwurfsmerkmale dar, die er für die Erstellung des Prozessmodells (EPK) für notwendig erachtet.*

**Neues Fallbeispiel erzeugen (Schritt 2):** Im nächsten Schritt erzeugt der Benutzer ein neues Fallbeispiel (Query-Case), das die Entwurfsmerkmale in der *Query* und einen leeren Solution-Teil enthält [Kram98, 89].

***Beispiel 5.3.2.2***

*Es ergibt sich somit der folgende Query-Case:*

$c_{query} = (Query, Solution = \phi)$

**Elementsuche (Schritt 3):** Nachdem die Entwurfsmerkmale ausgewählt wurden, kann die Fallbibliothek nun nach gleichen oder ähnlichen Fällen durchsucht werden. Dazu wird das Entwurfsmerkmal an der i-ten Stelle als Input und die lokale Ähnlichkeit und die aktivierten Fallbeispiele, die den aktivierten Knoten enthalten, als Output geliefert. [Kram98, 90]

Der Benutzer wird in der Auswahl, Interpretation und Untersuchung der Anwendbarkeit durch diesen Ansatz mit folgenden Funktionalitäten unterstützt [Kram98, 92]:

- Durch Indexbäume werden eine gezielte Suche und das Navigieren durch die Bibliothek unterstützt.
- Es können die Eigenschaften der Entwurfselemente gespeichert werden. Dazu zählt auch der Anwendungskontext. Außerdem ist es möglich die Verwendung eines Elements zu untersuchen (beispielsweise durch Verzweigung einer Funktion auf Entitätstypen oder auf Funktionsbäume und Prozessketten)
- Durch die hierarchische Darstellung der Modelle und Komponenten (Modell- und Komponentenbäume) können auch Elementvarianten identifiziert werden [Kram98, 154]. Dies erleichtert die Auswahl von Fallbeispielen durch den Benutzer. Die Organisation der Bäume ermöglicht auch die Berechnung von Ähnlichkeitsmaßen.

Das Durchsuchen der Fallbibliothek, die wie bereits erwähnt als semantisches Netz aufgebaut ist, erfolgt durch die Verwendung der Prolog-Eigenschaften des *Pattern Matching* und der *SICStus Objects.* SICStus Objects ist eine Spracherweiterung von *SICStus Prolog*, die eine flexible, objektorientierte Datenstrukturierung ermöglicht. Diese eignet sich besonders für die Wiederverwendung großer Anwendungen. [Kram98, 129ff]

Durch die Verwendung dieser Sprachen können laut Krampe folgende Eigenschaften ausgenutzt werden [Kram98, 153]:

- Durch den Verweis der Komponenten auf Modelle und von diesen auf Fallbeispiele können die geeigneten Modelle und Fallbeispiele gefunden werden.
- Durch die Zuweisung der Entwurfselemente auf Indexkategorien können auch ähnliche Elementvarianten gefunden werden. Es werden die Vererbungsbeziehungen zwischen Entwurfselementen und Indexkategorien untersucht.

- Es können spezialisierte Elemente durch Ableitung von Entwurfselementen entstehen. Ähnliche Elementvarianten können durch die Vererbungsbeziehung zwischen den Elementen bestimmt werden.

Anhand des oben erwähnten Ablaufs der Retrieve-Phase werden mit Hilfe dieser Eigenschaften jene Fallbeispiele aktiviert und als Source-Cases identifiziert, die die Entwurfsmerkmale und ähnliche Elementvarianten enthalten. Die lokale Ähnlichkeitsbestimmung wird bei der Untersuchung außen vor gelassen, da auf eine Erstellung des semantischen Netzes als Organisationsbasis verzichtet wurde.

**Ähnlichkeit bestimmen (Schritt 4):** Die von Krampe verwendete Ähnlichkeitsbestimmung wird für die Untersuchung dieses Ansatzes vereinfacht, da es den Rahmen dieser Diplomarbeit sprengen würde. Unter *lokaler Ähnlichkeit* meint Krampe den Vergleich zwischen dem Entwurfsmerkmal und den Entwurfselementen in der Entwurfsbibliothek, wobei in den Bäumen der Abstand zum *„speziellsten, gemeinsamen Vorgänger“* berechnet wird. *Globale Ähnlichkeit* bedeutet eine Summenbildung der lokalen Ähnlichkeit. [Kram98, 98ff]

Krampe rechtfertigt die Verwendung der Ähnlichkeitsbestimmung folgendermaßen:

*„Das Maß der Ähnlichkeit von Entwürfen soll einen Hinweis für den Aufwand der Wiederverwendung in einer gegebenen Problemsituation geben. Dazu wird der Query-Case mit den in der Fallbibliothek enthaltenen Fallbeispielen verglichen und deren Ähnlichkeit berechnet. Die theoretisch beste Wiederverwendbarkeit bietet das Fallbeispiel mit der größten Ähnlichkeit zum Source-Case.“* [Kram98, 96f]

Der Wiederverwendungsaufwand kann jedoch nur näherungsweise bestimmt werden, da sich die Nützlichkeit einer Komponente oder eines Entwurfs erst während des Entwurfsprozesses herausstellt. Desweiteren haben Informationssystementwürfe eine komplexe Struktur und die Ähnlichkeit kann nur schwierig festgestellt werden. Da für die Lösung des Entwurfsproblems mehrere Teilentwürfe benötigt werden, bedeutet dies einen zusätzlichen Anpassungsaufwand. Trotz dieser Bedenken ist jedoch die Bestimmung des Ähnlichkeitsmaßes laut Krampe durchaus sinnvoll, um die für die Lösung relevanten Fallbeispiele zu bestimmen. [Kram98, 97]

***Beispiel 5.3.2.3***

> *Ausgehend von der zuvor ausgewählten Query werden iterativ alle Entwurfsmerkmale mit den Entwurfselementen in der Entwurfsbibliothek mit Hilfe von Pattern Matching verglichen. Das erste Entwurfsmerkmal in der Query ist die Funktion ’Kreditrisiko/Bonität prüfen’. Vergleicht man diese Funktion mit den Fallbeispielen in der Fallbibliothek, ist sie in den Prozessketten ‚Kreditvergabe mit Bonitätsprüfung, Risiko hoch/mittel/niedrig’ (model_100) und ‚Kreditvergabe mit Bonitätsprüfung, Risiko niedrig/hoch’ (model_101) enthalten. Diese Modelle sind in den Fallbeispielen case_1 und case_2 enthalten (vgl. Abbildung 114 und Abbildung 115). Werden nun alle Entwurfsmerkmale auf diese Art verglichen, werden folgende Source-Cases mit der angeführten globalen Ähnlichkeit identifiziert:*

| *Source-Cases* | *Globale Ähnlichkeit* |
|---|---|
| *case_1* | *5 $k_i$* |
| *case_2* | *4 $k_i$* |
| *case_3* | *7 $k_i$* |

*Die globale Ähnlichkeit gibt die Anzahl der enthaltenen Entwurfsmerkmale in den gefundenen Source-Cases an.*

Wie aus dem Beispiel ersichtlich ist, wurden die Source-Cases auf lokaler Ebene durch Pattern Matching verglichen und eine Identifizierung von ähnlichen Elementvarianten außer Acht gelassen.

### 5.3.3 Anwendung von gefundenen Prozesslösungen

Ziel dieser Phase ist eine erste Lösung des Problems. In der Retrieve-Phase (Abschnitt 5.3.2) wurden mehrere Source-Cases (Fallbeispiele) gefunden und durch die Ähnlichkeitsbestimmung jene identifiziert, die dem neuen Problem somit am ähnlichsten sind. [Kram98, 100]

In der Reuse-Phase muss jedoch die Auswahl spezifizierter durchgeführt werden, *„da Teile des Entwurfsproblem möglicherweise in anderen Source-Cases besser gelöst sind als im ähnlichsten.“* Daher werden in der Reuse-Phase die gefundenen Fallbeispiele zu einer optimalen Lösung zusammengeführt. Krampe identifiziert dazu die Teilaufgaben *Teilentwürfe bestimmen*, *Teilentwürfe wählen*, die *ausgewählten Teilentwürfe kopieren*, die *Entwurfsschemata konfigurieren* und *unbekannte Entwurfselemente lesen*. [Kram98, 90ff]

In dieser Phase wird als Input die *Query* geliefert, die in der Retrieve-Phase festgelegt wurde und in der $c_{query}$ gespeichert ist, alle aktivierten Fallbeispiele $c_{source}$ und alle Konsistenzregeln (vgl. bezüglich Konsistenzregeln weiter unten). Der Output dieser Phase sind angepasste Teilentwürfe, die logisch zusammengehören, über Regeln ausgewählt und angepassten Beziehungen zwischen den eingefügten Teilentwürfen, eventuell konfigurierte Entwurfsschemata und eingelesene, unbekannte Entwurfselemente, die noch nicht in den Bäumen gespeichert sind. [Kram98, 100ff] Die einzelnen Schritte werden nachfolgend detaillierter erklärt. Zu den Schritten 1, 2 und 3 werden der Input, der Output und die Datenstrukturen der verwendeten Elemente tabellarisch dargestellt werden. Zu den Schritten 4 und 5 werden die Algorithmen nicht dargestellt und werden daher nur natürlichsprachig erklärt.

Abbildung 118 zeigt den algorithmischen Ablauf der Reuse-Phase:

Schritt 1: Bestimme Teilentwürfe $E_j$:
    Für jedes $k_i \in c_{query}$:
        Bestimme Source-Cases, die $k_i$ enthalten: $sim(k_i, c_{source}) > 0$
        Identifiziere Teilentwürfe $E_j$, die zu $k_i$ gehören
        Bestimme $K_j := \{k \in \{k_1,\ldots,k_m\} | sim(k, E_j) > 0$
Schritt 2: Wähle Teilentwürfe $E_j$ aus, d.h. finde optimale Überdeckung von $c_{query}$:
    $K_{it} := \{ k_1,\ldots,k_m \}$, *Überdeckung* := 0
    Solange $K_{it} \neq 0$:
        Bestimme $E_j$ mit $sim(K_{it}, E_j)$ maximal

*Überdeckung = Überdeckung* $\cup \{j\}$
$K_{it} = K_{it} \setminus K_j$
Schritt 3: Kopiere ausgewählte Teilentwürfe:
Für jedes $j \in$ *Überdeckung*:
Kopiere $E_j$ zur Entwurfslösung $c_{query}$:
$c_{query} = c_{query} \cup E_j \setminus \{c_{query} \cap E_j\}$
Passe Beziehungen des kopierten an $E_j$
Schritt 4: Konfiguriere Entwurfsschemata
Schritt 5: Lese unbekannte Entwurfselemente

**Abbildung 118: Ablauf Reuse-Phase nach Krampe**

**Teilentwürfe bestimmen (Schritt 1):** In der ersten Teilaufgabe der Reuse-Phase werden Teilentwürfe bestimmt und in der Menge $E_j$ aufgenommen [Kram98, 156].

Der Input bei dieser Teilaufgabe sind die ausgewählten Entwurfselemente in der $c_{query}$, die in der in der Retrieve-Phase festgelegten *Query* gespeichert sind, alle aktivierten Fallbeispiele $c_{source}$ und alle Konsistenzregeln (vgl. bezüglich Konsistenzregeln weiter unten). Der Output besteht aus den erweiterten Mengen $E_j$ und $K_j$. In der Menge $E_j$ wird jener Teilentwurf gespeichert, der aus allen aktivierten Entwurfselementen besteht, die gemäß der verwendeten Konsistenzregeln aktivierten wurden. Zu den aktivierten Entwurfselementen in der Menge $E_j$ sind somit nicht nur die ausgewählten Entwurfsmerkmale enthalten, sondern auch Elemente gemäß der verwendeten Konsistenzregeln, beispielsweise gesamte Modelle (Entity-Relationship-Modell, Funktionsbaum, Prozesskette) und logisch zusammengehörende Komponenten daraus. In der Menge $K_j$ werden die gefundenen Elemente festgehalten, die in der *Query* enthalten sind. [Kram98, 101ff] In Tabelle 10 sind der Input, der Output und die Datenstruktur des Algorithmus aus Abbildung 119 übersichtlich dargestellt. Die einzelnen Schritte des Algorithmus werden nachfolgend detailliert erklärt.

| | |
|---|---|
| ***Input*** | $c_{query}$ |
| | $c_{source}$ |
| | Konsistenzregeln |
| ***Output*** | $E_j$ |
| | $K_j$ |
| ***Elemente und Datenstrukturen*** | |
| $k_i$… | Entwurfselement an der i-ten Stelle, kann ein Modell (z.B. EPK, Funktionsbaum oder Entity-Relationship-Modell) oder eine Komponente daraus sein (z.B. Funktion oder Ereignis) |
| $i$… | Index der Entwurfselemente, 1…m |
| $c_{query}$… | zu erstellendes Fallbeispiel, bestehend aus *Query* und leerer *Solution* |
| $c_{source}$… | gesamte Menge der Source-Cases, die mindestens ein Entwurfselement enthalten, Source-Cases bestehen ebenfalls aus *Query* und *Solution* |
| $E_j$… | Teilentwurf an der j-ten Stelle, Menge der aktivierten Modelle und/oder Modellkomponenten |
| $j$… | Index der Teilentwürfe, 1…n |
| $K_j$… | im Teilentwurf $E_j$ enthaltene Entwurfselemente aus der *Query*, Vereinigungsmenge der Entwurfselemente *k* |

| | |
|---|---|
| *k…* | Entwurfselement aus der *Query*, für das gilt: $sim(k, E_j) > 0$, Modell (z.B. EPK, Funktionsbaum oder Entity-Relationship-Modell) oder Modellkomponente |
| ***Hilfsfunktion*** | |
| $sim(k_i, c_{source})$ | Liefert als Rückgabewert die Summe der lokalen Ähnlichkeiten zwischen den Entwurfselementen $k_i$ und $c_{source}$. $c_{source}$ bezeichnet die gesamte Menge aller Source-Cases. |
| $sim(k, E_j)$ | Liefert als Rückgabewert die Summe der lokalen Ähnlichkeiten zwischen den Entwurfselementen *k* und den Entwurfselementen aus $E_j$. |
| *Konsistenzregeln* | Beschreibung der Bedingungen, sodass ein Modell bzw. eine Modellkomponente als konsistent anzusehen ist. Durch die Anwendung der betroffenen Konsistenzregel werden zusätzliche Entwurfselemente in den Teilentwurf $E_j$ aufgenommen, die gemäß der Metamodellkonsistenz- oder Anwendungsdomänenkonsistenzregel eine logische Einheit bilden (vgl. weiter unten) |

**Tabelle 10: Algorithmenbeschreibung zu Abbildung 119**[11]

Abbildung 119 zeigt den Ablauf dieser Teilaufgabe algorithmisch wie folgt:

Bestimme Teilentwürfe $E_j$:
    Für jedes $k_i \in c_{query}$:
        Bestimme Source-Cases, die $k_i$ enthalten: $sim(k_i, c_{source}) > 0$
        Für jede Konsistenzregel, von der $k_i$ betroffen ist:
            Aktiviere alle Entwurfselemente, die im Bedingungsteil vorkommen
        Aktivierte Elemente bilden $E_j$
        Bestimme $K_j := \{k \in \{k_1,\ldots,k_m\} | sim(k, E_j) > 0$

**Abbildung 119: Teilaufgabe "Identifikation von Teilentwürfen" [Kram98, 156]**

Zuerst soll geprüft werden, ob logisch zusammengehörende Teilentwürfe gefunden werden können. Würde das nicht gemacht werden, würde eine nicht zusammenhängende Lösung entstehen und der weitere Entwurfsprozess dadurch erschwert werden. [Kram98, 102]

Die erste Teilaufgabe wird wie folgt gelöst: *„Ein ausgewähltes Element $k_i$ induziert zu jedem Fallbeispiel, in dem es enthalten ist, einen Teilentwurf $E_j$. Daher werden bei der Bestimmung der Teilentwürfe zunächst die Source-Cases bestimmt, die ein $k_i$ enthalten und dann zu jedem Source-Case ein Teilentwurf $E_j$ identifiziert.“* Dabei wird auf die *Metamodellkonsistenzregeln*, die *Anwendungsdomänekonsistenzregeln* und das *Qualifizieren semantischer Beziehungen* zurückgegriffen. Die Konsistenzregeln bestimmen für welches aktivierte Element welche zusätzlichen Elemente aktiviert werden sollen. [Kram98, 102]

Die *Metamodellkonsistenzregeln* definieren die zu aktivierenden Elemente für Funktionen, Entitätstypen, Ereignisse, Attribute, Funktionsbäume, ER-Modelle und Prozessketten. Beispielsweise müssen bei einer aktivierten Funktion die die Funktion enthaltende Prozesskette, die Daten, die von der Funktion gelesen oder geschrieben werden, die von der Funktion ausgelösten Ereignisse und die von der Funktion erzeugten Ereignisse aktiviert werden. Diese Regeln werden über die Metamodelle der verwendeten Modellierungsmethode bestimmt [Kram98, 102ff].

[11] Algorithmenbeschreibung samt den Datenstrukturen wurde aus [Kram98, 89ff] entnommen.

Die *Anwendungsdomänekonsistenzregeln* (auch Referenzmodellregeln) besagen, welche Elemente zu einer strukturellen Einheit gehören. Krampe führt für diese Konsistenzregel das Beispiel „Entitätstyp Lieferschein“ an. Hier müssen die sowohl die Funktionen und Prozessketten, die den Lieferantenstamm verwalten, die *„Funktionen und Prozessketten zum Bestellwesen“*, die *„Funktionen und Prozessketten zum Wareneingang“* und das *„ER-Modell für Artikel“* aktiviert werden. Es können jedoch auch Elemente zu Elementkategorien zusammengefasst werden, wie z.B. Stammdaten. [Kram98, 102ff] Die Referenzmodellregeln werden durch den Import eines Referenzmodells bestimmt [Kram98, 116f].

In dieser Arbeit werden Funktionen als Entwurfsmerkmale verwendet. Daher werden folgende Metamodellkonsistenzregeln für Funktionen in Anlehnung an [Kram98, 161] formuliert und leicht abgeändert, da Funktionsbäume und Entity-Relationship-Modelle und daher Entitätstypen ebenfalls nicht verwendet werden. Folgendes Beispiel stellt dies dar.

***Beispiel 5.3.3.1***

Funktionskonsistenzregel:
Funktion *f* konsistent ←
∃ Prozessketten *p*, die *f* enthalten ∧
∃ Daten *d*, die von *f* gelesen/geschrieben werden ∧
∃ Ereignisse *e*, die *f* auslösen ∧
∃ Ereignisse *e*, die *f* erzeugt

*Diese Konsistenzregel besagt, dass eine Funktion dann konsistent ist, wenn mindestens eine Prozesskette die Funktion enthält, mindestens ein Datentyp von der Funktion gelesen oder geschrieben wird, mindestens ein Ereignis die Funktion auslöst und mindestens ein Ereignis von der Funktion erzeugt wird.*

Somit wird gewährleistet, dass die benachbarten Elemente kontrolliert in die Entwurfslösung aufgenommen und nicht das gesamte semantische Netz aktiviert wird und eine Anpassung an das neue Entwurfsproblem erschweren würde [Kram98, 102f].

Zusätzlich zu den oben erwähnten Konsistenzregeln können noch semantische Beziehungen qualifiziert werden. Dies ist dann sinnvoll, wenn die Regeln nicht alle zur Wiederverwendung notwendigen benachbarten Knoten aktivieren würden. Hierzu werden Beziehungen Relevanzgrade zugeordnet. Wenn die Beziehungsrelevanz *„einen bestimmten Schwellwert übersteigt, wird der benachbarte Knoten aktiviert. Auf diese Weise können Elementfamilien gebildet werden wie beispielsweise eine logische Folge von Prozessketten.“* [Kram98, 103f]

Gemäß Krampe sollen diese drei Strategien gemeinsam angewendet werden. Dadurch können einzelne Elemente zwar öfter aktiviert werden. Dennoch sollen diese Strategien verwendet werden, um identifizierte Teilentwürfe möglichst konsistent und vollständig zu halten. [Kram98, 104]

Im Rahmen dieser Diplomarbeit werden die *Anwendungsdomänekonsistenzregeln* und das *Qualifizieren semantischer Beziehungen* vernachlässigt, da die Rahmenbeispiele für die Untersuchung klein gehalten werden und die *Metamodellkonsistenzregeln* ausreichen.

***Beispiel 5.3.3.2***

*Gemäß der Vorgehensweise nach Krampe werden zuerst für jedes $k_i$ aus $c_{query}$ die Source-Cases bestimmt, die mindestens ein $k_i$ enthalten. Zuerst wird hier 'Kreditrisiko/Bonität prüfen' dargestellt. Die Source-Cases, die $k_1$ mindestens einmal enthalten sind case_1 und case_2:*

$$k_1 := \{\ \text{'Kreditrisiko/Bonität prüfen'}\ \}$$

Source-Cases, die $k_1$ enthalten $sim(k_1, c_{source}) > 0$

$$c_{source} := \{\ case\_1, case\_2\ \}$$

*Somit müssen die Teilentwürfe für case_1 und case_2 bestimmt werden.*

***Case_1:***

*Dazu müssen die aktivierten und die zu aktivierenden Entwurfselemente in $E_1$ gemäß der Konsistenzregel für Funktionen (vgl. weiter oben) aufgenommen werden.*

| |
|---|
| Für jede Konsistenzregel, von der $k_1$ betroffen ist:<br>Aktiviere alle Entwurfselemente, die im Bedingungsteil vorkommen<br>Aktivierte Elemente bilden $E_1$ |

*Das aktivierte Entwurfselement ist die Funktion $k_1$ und die zu aktivierenden Entwurfselemente sind gemäß der Funktionskonsistenzregel die Prozesskette, die $k_1$ enthält, die auslösenden und erzeugten Ereignisse und die Informationsobjekte als Daten. Das erste Element in $E_1$ wäre somit die Prozesskette aus Abbildung 6. Die Rahmenbeispiele werden für die Analyse klein gehalten. Aus diesem Grund werden nur alle anderen Elemente gemäß der Funktionskonsistenzregel in den Teilentwurf $E_1$ aufgenommen. Diese werden aus Abbildung 120 entnommen.*

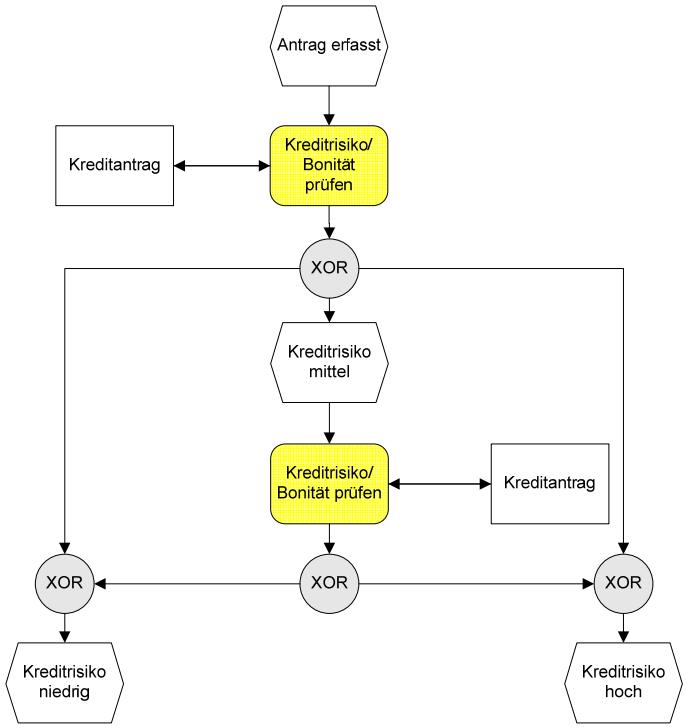



**Abbildung 120: Aktiviertes und zu aktivierende Entwurfselemente laut Krampe (case_1)[12]**

*$E_1$ beinhaltet das aktivierte Element $k_1$, die gelesenen und geschriebenen Daten (Informationsobjekte), die auslösenden und erzeugten Ereignisse. Es werden alle Elemente nur einmal in die Menge aufgenommen. Somit bildet $E_1$ folgende Menge:*

$$E_1 := \{\ \textbf{'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag',}$$
$$\textbf{'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditrisiko niedrig'}\ \}^{13}$$

Als nächstes wird für den Teilentwurf $E_j$ die Menge $K_j$ bestimmt. Die Menge $K_j$ besteht aus der Schnittmenge zwischen den ausgewählten Entwurfselementen und den Elementen des Teilentwurfs $E_j$. Der Teilentwurf ist für die endgültige Entwurfslösung umso geeigneter, *„je mehr Elemente dies sind“*. [Kram98, 104]

***Beispiel 5.3.3.3***

***Case_1:***
*$K_1$ wird nun das Entwurfsmerkmal $k_1$ hinzugefügt, da dies in dem Fallbeispiel case_1 enthalten ist. Somit ergibt sich für $K_1$:*

$$K_1 := \{\ \textbf{'Kreditrisiko/Bonität prüfen'}\ \}^{14}$$

***Case_2:***
*Da das Entwurfsmerkmal $k_1$ in case_2 ebenfalls enthalten ist, wird die zuvor durchgeführte Vorgehensweise auf dieses Fallbeispiel ebenfalls angewendet. Die gemäß*

[12] Aktiviertes Element ist hervorgehoben.
[13] Neu hinzugefügte Elemente sind fett dargestellt.
[14] Neu hinzugefügte Elemente sind fett dargestellt.

*der Funktionskonsistenzregel aktivierten und zu aktivierenden Entwurfselemente laut Krampe sind für dieses Fallbeispiel nachstehend abgebildet (vgl. Abbildung 121).*

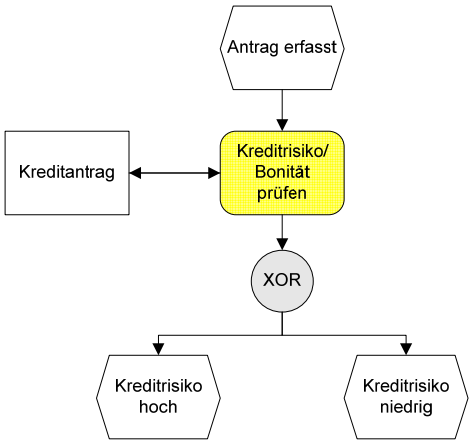



**Abbildung 121: Funktion „Kreditrisiko/Bonität prüfen“/case_2**[15]

*Analog zu case_1 ergibt sich für case_2 folgende Menge für den Teilentwurf $E_2$:*

$$E_2 := \{\ \textbf{\textit{'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko hoch', 'Kreditrisiko niedrig'}}\ \}$$

*$K_2$ wird nun das Entwurfsmerkmal $k_1$ hinzugefügt, da dies ebenfalls in dem Fallbeispiel case_2 enthalten ist. Somit ergibt sich für $K_2$:*

$$K_2 := \{\ \textbf{\textit{'Kreditrisiko/Bonität prüfen'}}\ \}$$

*Diese Vorgehensweise wird nun für alle in der $c_{query}$ enthaltenen Entwurfsmerkmale $k_i$ durchgeführt. Auf eine wiederholte Erklärung der Vorgehensweise wird verzichtet und der Interpretation des Lesers überlassen.*

$$k_2 = \{\ \textit{Kreditantrag ablehnen}\ \}$$

Source-Cases, die $k_2$ enthalten $sim(k_2, c_{source}) > 0$

$$c_{source} = \{\ \textit{case\_1,case\_2}\ \}$$

***Case_1:***

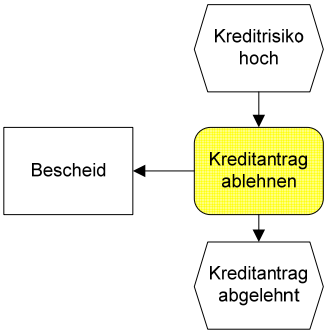



**Abbildung 122: Funktion "Kreditantrag ablehnen"/case_1**

[15] Aktiviertes Element ist hervorgehoben.

$E_1 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch',* ***'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt',*** *'Kreditrisiko niedrig'* $\}$

$K_1 := \{$ *'Kreditrisiko/Bonität prüfen',* ***'Kreditantrag ablehnen'*** $\}$

***Case_2:***

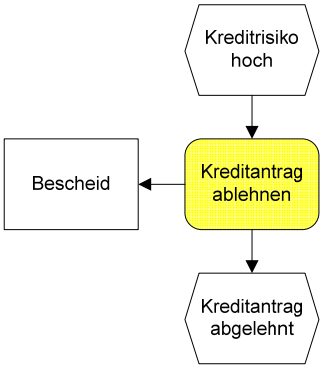



**Abbildung 123: Funktion "Kreditantrag ablehnen"/case_2**

$E_2 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko hoch',* ***'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt',*** *'Kreditrisiko niedrig'* $\}$

$K_2 := \{$ *'Kreditrisiko/Bonität prüfen',* ***'Kreditantrag ablehnen'*** $\}$

$k_3 = \{$ *'Kreditantrag bewilligen'* $\}$

Source-Cases, die $k_3$ enthalten $sim(k_3, c_{source}) > 0$

$c_{source} := \{$ *case_1,case_2* $\}$

***Case_1:***

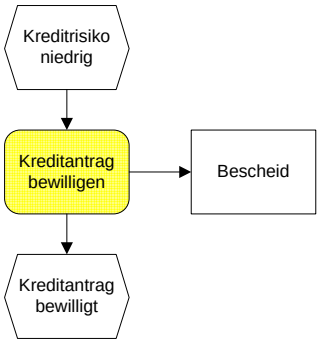



**Abbildung 124: Funktion "Kreditantrag bewilligen"/case_1**

$E_1 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig',* ***'Kreditantrag bewilligen', 'Kreditantrag bewilligt'*** $\}$

$K_1 := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen',* ***'Kreditantrag bewilligen'*** $\}$

***Case_2:***

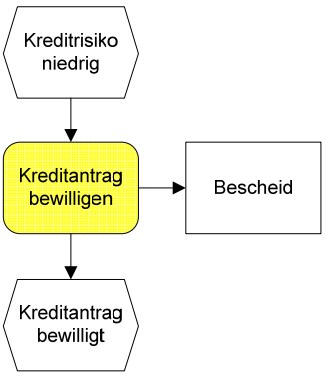



**Abbildung 125: Funktion "Kreditantrag bewilligen"/case_2**

$E_2 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig',* ***'Kreditantrag bewilligen', 'Kreditantrag bewilligt'*** $\}$

$K_2 := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen',* ***'Kreditantrag bewilligen'*** $\}$

$k_4 = \{$ *'Kreditvertrag erstellen'* $\}$

Source-Cases, die $k_4$ enthalten $sim(k_4, c_{source}) > 0$

$c_{source} := \{$ *case_1,case_2* $\}$

***Case_1:***

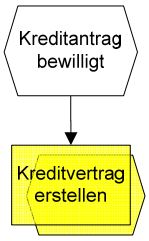



**Abbildung 126: Prozesspfad "Kreditvertrag erstellen"/case_1**

$E_1 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt',* ***'Kreditvertrag erstellen'*** $\}$

$K_1 := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen',* ***'Kreditvertrag erstellen'*** $\}$

***Case_2:***

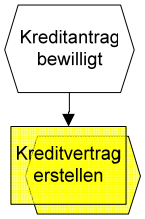



**Abbildung 127: Prozesspfad "Kreditvertrag erstellen"/case_2**

$E_2 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt',* ***'Kreditvertrag erstellen'*** $\}$

$K_2 := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen',* ***'Kreditvertrag erstellen'*** $\}$

$k_5 = \{$ *'Kunde informieren'* $\}$

Source-Cases, die $k_5$ enthalten $sim(k_5,c_{source}) > 0$

$c_{source} := \{$ *case_1* $\}$

***Case_1:***

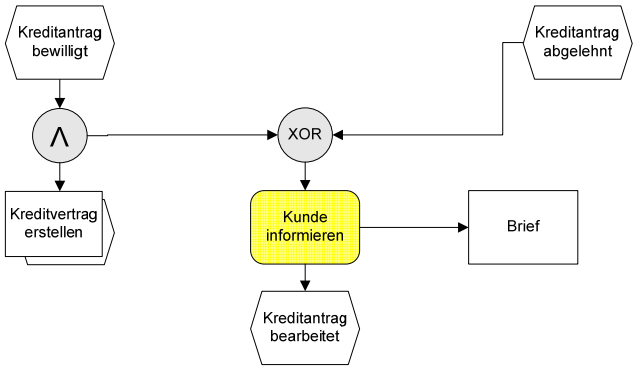



**Abbildung 128: Funktion "Kunde informieren"/case_1**

$E_1 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt', 'Kreditvertrag erstellen',* ***'Kunde informieren', 'Brief', 'Kreditantrag bearbeitet'*** $\}$

$K_1 := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen',* ***'Kunde informieren'*** $\}$

$k_6 = \{$ *'Grundbuchauszug beschaffen'* $\}$

Source-Cases, die $k_6$ enthalten $sim(k_6,c_{source}) > 0$

$c_{source} := \{$ *case_3* $\}$

***Case_3:***

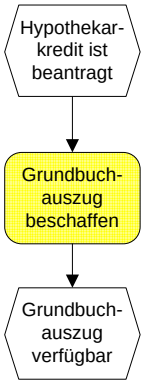



**Abbildung 129: Funktion "Grundbuchauszug beschaffen"/case_3**

$E_3 := \{$ ***'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen',***
***'Grundbuchauszug verfügbar'*** $\}$

$K_3 := \{$ ***'Grundbuchauszug beschaffen'*** $\}$

$k_7 = \{$ *'Bewertungsgutachten beschaffen'* $\}$

Source-Cases, die $k_7$ enthalten $sim(k_7,c_{source}) > 0$

$c_{source} := \{$ *case_3* $\}$

***Case_3:***

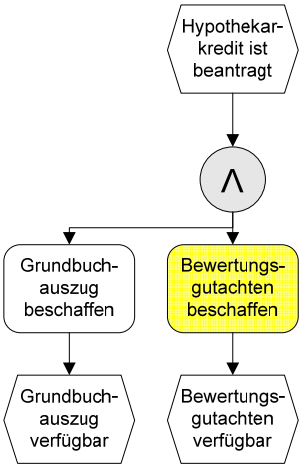



**Abbildung 130: Funktion "Bewertungsgutachten beschaffen"/case_3**

$E_3 := \{$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen',*
*'Grundbuchauszug verfügbar',* ***'Bewertungsgutachten beschaffen',***
***'Bewertungsgutachten verfügbar'*** $\}$

$K_3 := \{$ *'Grundbuchauszug beschaffen',* ***'Bewertungsgutachten beschaffen'*** $\}$

$k_8 = \{$ *'Einkommensnachweis beschaffen'* $\}$

Source-Cases, die $k_8$ enthalten $sim(k_8,c_{source}) > 0$

$c_{source} := \{$ *case_3* $\}$

***Case_3/model_1021 (a) und model_1022 (b):***

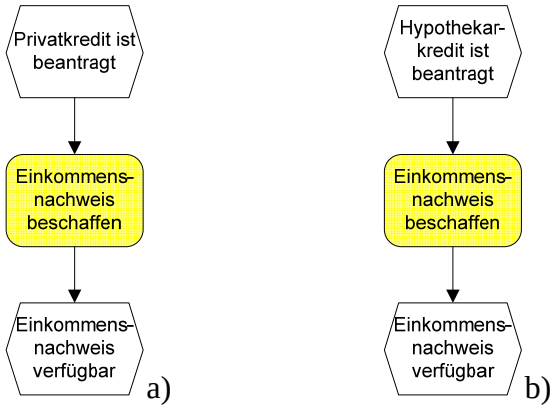



**Abbildung 131: Funktionen "Einkommensnachweis beschaffen"/case_3**

$E_3 := \{$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar',* ***'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar'*** $\}$

$K_3 := \{$ *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen',* ***'Einkommensnachweis beschaffen'*** $\}$

$k_9 = \{$ *'Beleihungsgrenze ermitteln'* $\}$

Source-Cases, die $k_9$ enthalten $sim(k_9, c_{source}) > 0$

$c_{source} := \{$ *case_3* $\}$

***Case_3/ model_1022:***

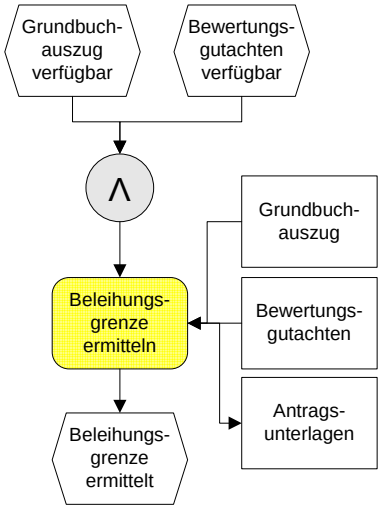



**Abbildung 132: Funktion "Beleihungsgrenze ermitteln"/case_3**

$E_3 := \{$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar',*

***'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt'*** }

$K_3 :=$ { *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen',* ***'Beleihungsgrenze ermitteln'*** }

$k_{10} =$ { *'Max. Kredithöhe berechnen'* }

Source-Cases, die $k_{10}$ enthalten $sim(k_{10}, c_{source}) > 0$

$c_{source} :=$ { *case_3* }

***Case_3/model_1021 (a) und model_1022 (b):***

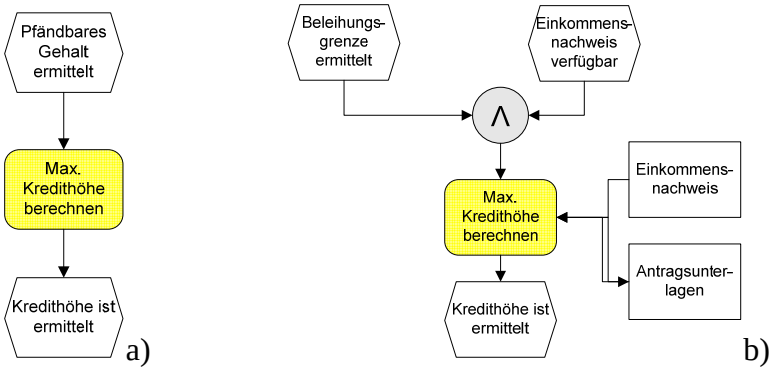



**Abbildung 133: Funktionen "Max. Kredithöhe berechnen"/case_3**

$E_3 :=$ { *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar', 'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt',* ***'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt'*** }

$K_3 :=$ { *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln',* ***'Max. Kredithöhe berechnen'*** }

$k_{11} =$ { *'Insolvenzauskunft beschaffen'* }

Source-Cases, die $k_{11}$ enthalten $sim(k_{11}, c_{source}) > 0$

$c_{source} :=$ { *case_3* }

***Case_3:***

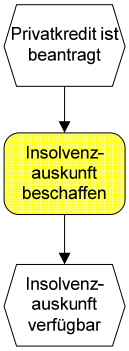



**Abbildung 134: Funktion "Insolvenzauskunft beschaffen"/case_3**

$E_3 := \{$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar',* ***'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar',*** *'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt', 'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt'* $\}$

$K_3 := \{$ *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen',* ***'Insolvenzauskunft beschaffen'*** $\}$

$k_{12} = \{$ *'Pfändbares Gehalt ermitteln'* $\}$

Source-Cases, die $k_{12}$ enthalten $sim(k_{11}, c_{source}) > 0$

$c_{source} := \{$ *case_3* $\}$

***Case_3:***

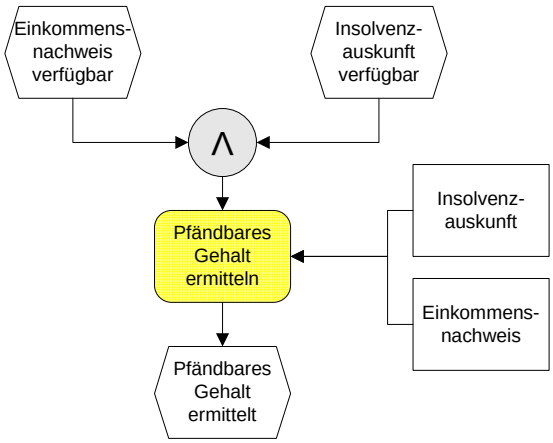



**Abbildung 135: Funktion "Pfändbares Gehalt ermitteln"/case_3**

$E_3 := \{$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar', 'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar',*

*'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt',* ***'Pfändbares Gehalt ermitteln', 'Insolvenzauskunft',*** *'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt'* }

$K_3 :=$ { *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen',* ***'Pfändbares Gehalt ermitteln'*** }

*Die nach der Durchführung der ersten Teilaufgabe vorhandenen Teilentwürfe sind demnach:*

*Die Menge für den Teilentwurf $E_1$ ist:*

$E_1 :=$ { *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt', 'Kreditvertrag erstellen', 'Kunde informieren', 'Brief', 'Kreditantrag bearbeitet'* }

*Die in case_1 enthaltenen Entwurfsmerkmale sind in $K_1$ abgebildet:*

$K_1 :=$ { *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren'* }

*Die Menge für den Teilentwurf $E_2$ ist:*

$E_2 :=$ { *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt', 'Kreditvertrag erstellen'* }

*Die in case_2 enthaltenen Entwurfsmerkmale sind in $K_2$ abgebildet:*

$K_2 :=$ { *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen'* }

*Die Menge für den Teilentwurf $E_3$ ist:*

$E_3 :=$ { *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar', 'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar', 'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt', 'Pfändbares Gehalt ermitteln', 'Insolvenzauskunft', 'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt'* }

*Die in case_3 enthaltenen Entwurfsmerkmale sind in $K_3$ abgebildet:*

$$K_3 := \{ \text{'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'} \}$$

*Es ist jener Teilentwurf für die Wiederverwendung geeigneter, je mehr Elemente in $K_j$ sind. Das angeführte Beispiel liefert $E_3$ als den Teilentwurf zurück, der am besten geeignet ist, da $K_3$ die meisten Entwurfselemente enthält.*

**Teilentwürfe auswählen (Schritt 2):** Im nächsten Schritt der Reuse-Phase sollen jene Teilentwürfe ausgewählt werden, bei denen der geringste Anpassungsaufwand zu erwarten ist. Aus diesem Grund soll versucht werden möglichst wenige Teilentwürfe zu verwenden und so eine *„Überlappung der ausgewählten Teilentwürfe"* zu minimieren. [Kram98, 104]

Der Input bei dieser Teilaufgabe sind die in der vorigen Teilaufgabe identifizierten Teilentwürfe $E_j$, die in diesem Teilentwurf $E_j$ vorhandenen Entwurfselemente in $K_j$ und die gesamte Menge der ausgewählten Entwurfselemente aus der $c_{query}$ (*Query*), die in der Menge $K_{it}$ festgehalten wird. Der Output besteht aus den Indizes jener Teilentwürfe, die die optimale Überdeckung liefern und in der Menge *Überdeckung* gespeichert sind, wobei zuerst immer jener ausgesucht wird, der die meisten Entwurfselemente aufweist. Dazu wird die Menge $K_{it}$ mit der Menge $E_j$ verglichen. [Kram98, 104] In Tabelle 11 sind der Input, der Output und die Datenstruktur des Algorithmus aus Abbildung 136 übersichtlich dargestellt. Die einzelnen Schritte des Algorithmus werden nachfolgend genau erklärt.

| | |
|---|---|
| ***Input*** | $c_{query}$ |
| | $E_j$ |
| | $K_j$ |
| ***Output*** | *Überdeckung* |
| ***Elemente und Datenstrukturen*** | |
| $c_{query}$… | zu erstellendes Fallbeispiel, bestehend aus *Query* und leerer *Solution* |
| $K_{it}$… | Menge der ausgewählten Entwurfselemente (*Query*), die noch durch Teilentwürfe $E_j$ abzudecken sind. |
| *Überdeckung…* | Menge der Indizes *j* der ausgewählten Teilentwürfe |
| $E_j$… | Teilentwurf an der j-ten Stelle, Menge der aktivierten Modelle und/oder Modellkomponenten |
| *j*… | Index der Teilentwürfe, 1…n |
| $K_j$… | Menge, der im Teilentwurf $E_j$ enthaltene Entwurfselemente aus der *Query*, Vereinigungsmenge der Entwurfselemente *k* |
| ***Hilfsfunktion*** | |
| $sim(K_{it},E_j)$ | Liefert als Rückgabewert die Summe der lokalen Ähnlichkeiten zwischen den Entwurfselementen aus der Menge $K_{it}$ und $E_j$. |

**Tabelle 11: Algorithmenbeschreibung zu Abbildung 136[16]**

[16] Algorithmenbeschreibung samt den Datenstrukturen wurde aus [Kram98, 89ff] entnommen.

Diese Teilaufgabe wird wie in Abbildung 136 dargestellt durchgeführt [Kram98, 101]:

Wähle Teilentwürfe $E_j$ aus, d.h. finde optimale Überdeckung von $c_{query}$:
$K_{it} := \{ k_1,\ldots,k_m \}$, *Überdeckung* $:= 0$
Solange $K_{it} \neq 0$:
Bestimme $E_j$ mit $sim(K_{it}, E_j)$ maximal
*Überdeckung* = *Überdeckung* $\cup \{j\}$
$K_{it} = K_{it} \setminus K_j$

**Abbildung 136: Teilaufgabe "Teilentwürfe auswählen" [Kram98, 101]**

Bei dieser Strategie wird die Auswahl der Teilentwürfe iterativ realisiert. Dabei wird bei jedem Iterationsschritt jener Teilentwurf bevorzugt, der die meisten Entwurfsmerkmale enthält. Bei jedem weiteren Schritt wird unter den restlichen Teilentwürfen *„derjenige ausgesucht, der die meisten bisher noch nicht abgedeckten Merkmale enthält“*. Die Iteration ist zu Ende, wenn alle Entwurfsmerkmale durch die ausgewählten Teilentwürfe abgedeckt sind. *„Die Menge $K_{it}$ gibt in jedem Iterationsschritt an, welche Merkmale durch die Teilentwürfe noch abzudecken sind, während Überdeckung sammelt, welche Teilentwürfe ausgewählt wurden. Da bei der Identifikation in den Mengen $K_j$ festgehalten wurde, welche Merkmale jeder Teilentwurf abdeckt, wird die Auswahl des besten Teilentwurfs erleichtert“*. Die gestellte Forderung mit möglichst wenigen Teilentwürfen auszukommen und daher Überlappungen zu verringern kann eventuell nicht zur Gänze erfüllt werden. Er begründet dies durch ein kombinatorisches Problem *„bei der Suche nach einer optimalen Überdeckung der Merkmale“*. Daher muss *„ein Kompromiss zwischen Rechenaufwand und optimaler Lösung eingegangen werden“*. [Kram98, 104f]

***Beispiel 5.3.3.4***

*Der Menge $K_{it}$ werden die ausgewählten Entwurfselemente hinzugefügt. Somit ist die Menge $K_{it}$ gleich der weiter oben angeführten Query (vgl. Beispiel 5.3.2.2, Seite 151). Die Überdeckung ist zu Beginn noch eine leere Menge:*

$K_{it} := \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren', 'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'* $\}$, *Überdeckung* $:= 0$

*Solange die Menge $K_{it}$ ungleich 0 ist, wird nun der Teilentwurf mit der maximalen Übereinstimmung ausgewählt und in die Menge der Überdeckung aufgenommen. Jene Elemente, die nun durch den Teilentwurf abgedeckt werden, werden aus der Menge $K_{it}$ entfernt.*
*Nun werden jene Teilentwürfe bestimmt, die die maximale Überdeckung aufweisen. Die Entwurfsmerkmale, die in der Menge $K_{it}$ und in dem Teilentwurf enthalten sind, werden fett dargestellt.*

$E_1 := \{$ *'Antrag erfasst',* ***'Kreditrisiko/Bonität prüfen'****, 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch',* ***'Kreditantrag ablehnen'****, 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig',* ***'Kreditantrag bewilligen'****,*

*'Kreditantrag bewilligt', **'Kreditvertrag erstellen', 'Kunde informieren',** 'Brief', 'Kreditantrag bearbeitet' }*

$E_2 :=$ { *'Antrag erfasst', **'Kreditrisiko/Bonität prüfen'**, 'Kreditantrag', 'Kreditrisiko hoch', **'Kreditantrag ablehnen'**, 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', **'Kreditantrag bewilligen'**, 'Kreditantrag bewilligt', **'Kreditvertrag erstellen'** }*

$E_3 :=$ { *'Hypothekarkredit ist beantragt', **'Grundbuchauszug beschaffen'**, 'Grundbuchauszug verfügbar', **'Bewertungsgutachten beschaffen'**, 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', **'Einkommensnachweis beschaffen'**, 'Einkommensnachweis verfügbar', **'Insolvenzauskunft beschaffen'**, 'Insolvenzauskunft verfügbar', **'Beleihungsgrenze ermitteln'**, 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt', **'Pfändbares Gehalt ermitteln'**, 'Insolvenzauskunft', 'Pfändbares Gehalt ermittelt', **'Max. Kredithöhe berechnen'**, 'Einkommensnachweis', 'Kredithöhe ist ermittelt' }*

*Als Ergebnis wird hier ebenfalls $E_3$ mit der maximalen Überdeckung zurückgeliefert (vgl. Menge $K_3$ in Beispiel 5.3.3.3, die die meisten Entwurfsmerkmale enthält). Im nächsten Schritt werden die Entwurfselemente dieses Teilentwurfs in die Überdeckung eingefügt.*

$$Überdeckung = Überdeckung \cup \{j\}$$

$$Überdeckung = 0 \cup \{\ 3\ \}$$

*Somit ergibt sich folgendes Ergebnis für die Überdeckung:*

| $Überdeckung = \{\ 3\ \}$ |
|---|

*Nun wird von der Menge $K_{it}$ die Menge aus $K_3$ abgezogen, um die Entwurfselemente herauszufinden, die noch von den verbleibenden Teilentwürfen abgedeckt werden müssen:*

$$K_{it} = K_{it} \setminus K_j$$

$K_{it} :=$ { *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren', 'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'* }

$K_3 :=$ { *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'* }

$K_{it} = \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren',* ***'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'*** $\} \setminus \{$ *'Grundbuchauszug beschaffen', 'Bewertungsgutachten beschaffen', 'Einkommensnachweis beschaffen', 'Beleihungsgrenze ermitteln', 'Max. Kredithöhe berechnen', 'Insolvenzauskunft beschaffen', 'Pfändbares Gehalt ermitteln'* $\}$[17]

*Die verbleibende Menge $K_{it}$ ist nun:*

$K_{it} = \{$ *'Kreditrisiko/Bonität prüfen', 'Kreditantrag ablehnen', 'Kreditantrag bewilligen', 'Kreditvertrag erstellen', 'Kunde informieren'* $\}$

*Analog zu dem ersten Iterationsschritt wird wieder die Schnittmenge von den Mengen $K_{it}$ und den Teilentwürfen eruiert und fett dargestellt:*

$E_1 := \{$ *'Antrag erfasst',* ***'Kreditrisiko/Bonität prüfen'****, 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch',* ***'Kreditantrag ablehnen'****, 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig',* ***'Kreditantrag bewilligen'****, 'Kreditantrag bewilligt',* ***'Kreditvertrag erstellen', 'Kunde informieren',*** *'Brief', 'Kreditantrag bearbeitet'* $\}$

$E_2 := \{$ *'Antrag erfasst',* ***'Kreditrisiko/Bonität prüfen'****, 'Kreditantrag', 'Kreditrisiko hoch',* ***'Kreditantrag ablehnen'****, 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig',* ***'Kreditantrag bewilligen'****, 'Kreditantrag bewilligt',* ***'Kreditvertrag erstellen'*** $\}$

*Somit wird der Teilentwurf $E_1$ ausgewählt, da mit diesem Entwurf fünf Entwurfsmerkmale abgedeckt werden. Es wird der Index dieses Teilentwurfs in die Überdeckung aufgenommen und die Elemente, die dieser Teilentwurf abdeckt aus der Menge $K_{it}$ entfernt:*

$Überdeckung = \{ 3,1 \}, K_{it} := 0$

*Da die Menge $K_{it}$ nun eine leere Menge ist, ist die Abbruchbedingung erfüllt und diese Teilaufgabe abgeschlossen. Es wurden somit zwei Teilentwürfe aus der Entwurfsbibliothek identifiziert, die die Forderung nach möglichst geringer Überlappung der Teilentwürfe" und „möglichst wenig Teilentwürfe" auszuwählen erfüllen.*

**Ausgewählte Teilentwürfe kopieren und anpassen (Schritt 3):** Nun können die ausgewählten Teilentwürfe zu einer optimalen Lösung zusammengesetzt werden [Kram98, 101].

[17] Die Überdeckungen werden fett dargestellt.

Input bei dieser Teilaufgabe sind die Überdeckung aus Schritt 2, die in Schritt 1 identifizierten Teilentwürfe $E_j$ und den in $c_{query}$ enthaltenen Entwurfselementen (*Query*). Der Output ist die gesamte Menge der kopierten Teilentwürfe und ihren angepassten Beziehungen untereinander, der in dem Solution-Teil der $c_{query}$ gespeichert wird. [Kram98, 105, 158] In Tabelle 12 sind wiederum der Input, der Output und die Elemente mit ihren Datenstrukturen übersichtlich dargestellt.

| | |
|---|---|
| ***Input*** | *Überdeckung* |
| | $E_j$ |
| | $c_{query}$ |
| ***Output*** | $c_{query}$ |
| ***Elemente und Datenstrukturen*** | |
| *j…* | Index der Teilentwürfe, 1…n |
| *Überdeckung…* | Menge der Indizes *j* der ausgewählten Teilentwürfe |
| $E_j$… | Teilentwurf an der j-ten Stelle, Menge der aktivierten Modelle und/oder Modellkomponenten |
| $c_{query}$… | zu erstellendes Fallbeispiel, bestehend aus *Query* und leerer *Solution* |

**Tabelle 12: Algorithmenbeschreibung zu Abbildung 137[18]**

Krampe schlägt auch hier wieder eine iterative Vorgehensweise vor, die wie folgt algorithmisch dargestellt wird:

Kopiere ausgewählte Teilentwürfe:
  Für jedes $j \in$ *Überdeckung*:
    Kopiere $E_j$ zur Entwurfslösung $c_{query}$:
      $c_{query} = c_{query} \cup E_j \setminus \{c_{query} \cap E_j\}$
    Passe Beziehungen des kopierten $E_j$ an

**Abbildung 137: Teilaufgabe "Kopieren von Teilentwürfen und Anpassung der Lösung" [Kram98, 101]**

[Kram98, 105] beschreibt den Ablauf dieser Aufgabe wie folgt: *„Als erstes wird der Teilentwurf in die Lösung kopiert, der über die meisten Entwurfsmerkmale verfügt. Bei jedem weiteren Kopiervorgang werden aus den hinzugefügten Teilentwürfen lediglich diejenigen Entwurfselemente kopiert, die die Lösung erweitern. In der Lösung bereits existierende Entwurfselemente werden belassen und die notwendigen Beziehungen zu den Elementen des jeweils neu hinzukommenden Teilentwurfs ersetzt. Nach dem Zusammenfügen der Teilentwürfe besteht die neue Entwurfslösung aus einer Liste von Funktionsbäumen, Entity-Relationship-Modellen und Prozessketten.“*

Die ausgewählten Entwurfsmerkmale der Teilentwürfe werden in den Solution-Teil kopiert und *„während des Kopiervorgangs“* werden die Kanten zwischen den Entwurfsmerkmalen und Prozessketten entsprechend des Anwendungskontextes angepasst [Kram98, 158], der aus der Aufgabenstellung aus Abschnitt 1.3.1 hervorgeht.

Der Algorithmus aus Abbildung 138 stellt den Algorithmus der Teilaufgabe aus Abbildung 137 genauer dar. Es wird der gleiche Input und Output verwendet und um Elemente und Datenstrukturen erweitert, die dem Vergleich von Modellen und ihren Modellkomponenten aus dem ausgewählten Teilentwurf $E_j$ mit den Entwurfselementen aus der *Query* dienen und

[18] Algorithmenbeschreibung samt den Datenstrukturen wurde aus [Kram98, 89ff] entnommen.

so die Beziehungen (Kanten) der in der $c_{query}$ enthaltenen Entwurfselemente belassen und nur neue Entwurfselemente hinzufügen [Kram98, 158]. In Tabelle 13 sind der Input, Output und die Datenstruktur des Algorithmus aus Abbildung 138 übersichtlich dargestellt. Die einzelnen Schritte des Algorithmus werden nachfolgend genau erklärt.

| | |
|---|---|
| ***Input*** | *Überdeckung* |
| | $E_j$ |
| | $c_{query}$ |
| ***Output*** | $c_{query}$ |
| ***Elemente und Datenstrukturen*** | |
| *j…* | Index der Teilentwürfe, 1…n |
| *Überdeckung…* | Menge der Indizes *j* der ausgewählten Teilentwürfe |
| $m_i$… | Modell mit Index i, $E_j$ besteht aus Modellen $m_i$ |
| $m'_i$… | Modellkopie, die nach $c_{query}$ kopiert wird |
| *i…* | Index der Modelle, 1…m |
| $E_j$… | Teilentwurf an der j-ten Stelle, Menge der aktivierten Modelle und/oder Modellkomponenten |
| $c_{query}$… | zu erstellendes Fallbeispiel, bestehend aus *Query* und leerer *Solution* |
| *k…* | Entwurfselement aus $m_i$, Modell (z.B. EPK, Funktionsbaum oder Entity-Relationship-Modell) oder Modellkomponente |
| *k'…* | Entwurfselement in $c_{query}$ in der *Solution* |
| ***Hilfsfunktion*** | |
| $sim_l(k,k')$ | Liefert als Rückgabewert ein Ähnlichkeitsmaß zwischen dem ausgewählten Entwurfselement *k* und dem Entwurfselement *k'*. *k'* bezeichnet ein ähnliches Entwurfselement, das bereits in der $c_{query}$ enthalten ist. Dabei wird der Abstand *„zum speziellsten, gemeinsamen Vorgänger“* berechnet. |

**Tabelle 13: Algorithmenbeschreibung zu Abbildung 138[19]**

Dazu zeigt folgende Vorgehensweise die Anpassung der Beziehungen detaillierter:

Für jedes $j \in$ *Überdeckung*[20]:
  Für jedes Modell $m_i \in E_j$:
    Kopiere $m_i$ nach $m_i$‘
    Für jede Komponente $k \in m_i$:
      Falls $\exists\, k' \in c_{query}$ mit $sim_l(k, k') > 0$:
        Ersetze in $m'_i$ alle *k*-Kanten durch *k‘*-Kanten
      Kopiere $m_i$‘ nach $c_{query}$

**Abbildung 138: Teilaufgabe „Anpassung der Entwürfe“ [Kram98, 158]**

Die in dem Teilentwurf $E_j$ enthaltenen Modelle $m_i$ werden kopiert. Um die Beziehungen anzupassen, wird für jede Komponente *k* untersucht, ob sie bereits in der Lösung enthalten ist. Wenn ja, müssen die Kanten, die mit *k* verknüpft sind, mit *k‘* verknüpft werden, um so

[19] Algorithmenbeschreibung samt den Datenstrukturen wurde aus [Kram98, 89ff] entnommen.

[20] Anstatt des Elements *Überdeckung* ist bei diesem Algorithmus in [Kram98, 158] das Element *Auswahl* angeführt. Diese beiden Elemente sind jedoch ident, da sie durch den gleichen Algorithmus befüllt werden (vgl. [Kram98, 157]).

die neuen Entwurfselemente hinzuzufügen und die alten Entwurfselemente in der $c_{query}$ zu belassen.

***Beispiel 5.3.3.5***

*Ausgehend von dem Ergebnis der vorherigen Teilaufgabe „Auswahl von Teilentwürfen“ (vgl. Beispiel 5.3.3.4) wird nun der Teilentwurf $E_3$ in den Solution-Teil der $c_{query}$ kopiert, da dieser die maximale Überdeckung liefert.*

*Im ersten Iterationsschritt werden somit alle aktivierten Elemente des Teilentwurfs $E_3$ in den Solution-Teil kopiert. Dies sind Komponenten der Prozessmodelle model_1021 und model_1022 (vgl. Abschnitt 1.4).*

$E_3 := \{$ ***'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar'***, *'Privatkredit ist beantragt',* ***'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar',*** *'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar',* ***'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt'****, 'Pfändbares Gehalt ermitteln', 'Insolvenzauskunft', 'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt'* $\}$

*Aus Prozessmodell model_1022 werden für den Teilentwurf $E_3$ folgende Entwurfselemente in den Solution-Teil der $c_{query}$ kopiert.*

$c_{query} = (Query, Solution =$ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar', 'Beleihungsgrenze ermitteln', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Beleihungsgrenze ermittelt')*

*Aus Prozessmodell model_1021 werden für den Teilentwurf $E_3$ folgende Entwurfselemente in den Solution-Teil der $c_{query}$ kopiert und die Beziehungen entsprechend der oben erwähnten Vorgehensweise angepasst.*

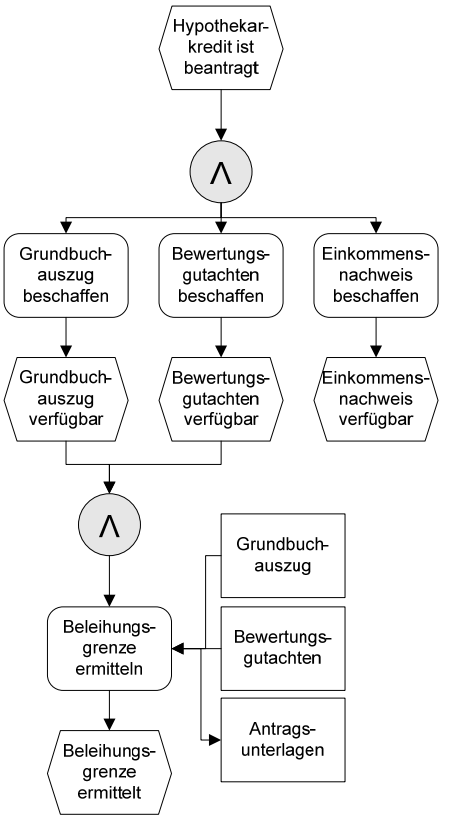



**Abbildung 139: Ausschnitt aus Prozessmodell model_1022**

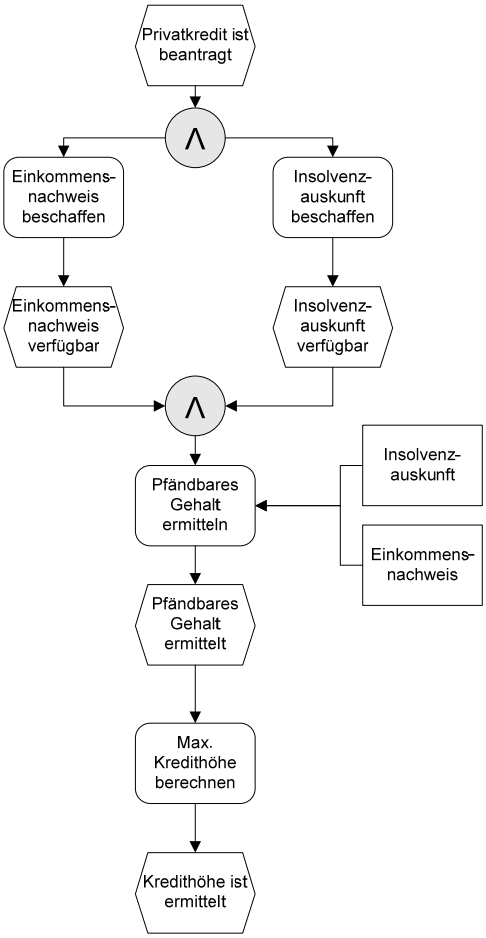



**Abbildung 140: Ausschnitt aus Prozessmodell model_1021**

$c_{query} = (Query, Solution = $ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis beschaffen', 'Einkommensnachweis verfügbar', 'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar', 'Beleihungsgrenze ermitteln', 'Beleihungsgrenze ermittelt', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Pfändbares Gehalt ermitteln', 'Insolvenzauskunft', 'Pfändbares Gehalt ermittelt', 'Max. Kredithöhe berechnen', 'Einkommensnachweis', 'Kredithöhe ist ermittelt')*

*Somit ergibt sich folgendes Prozessmodell:*

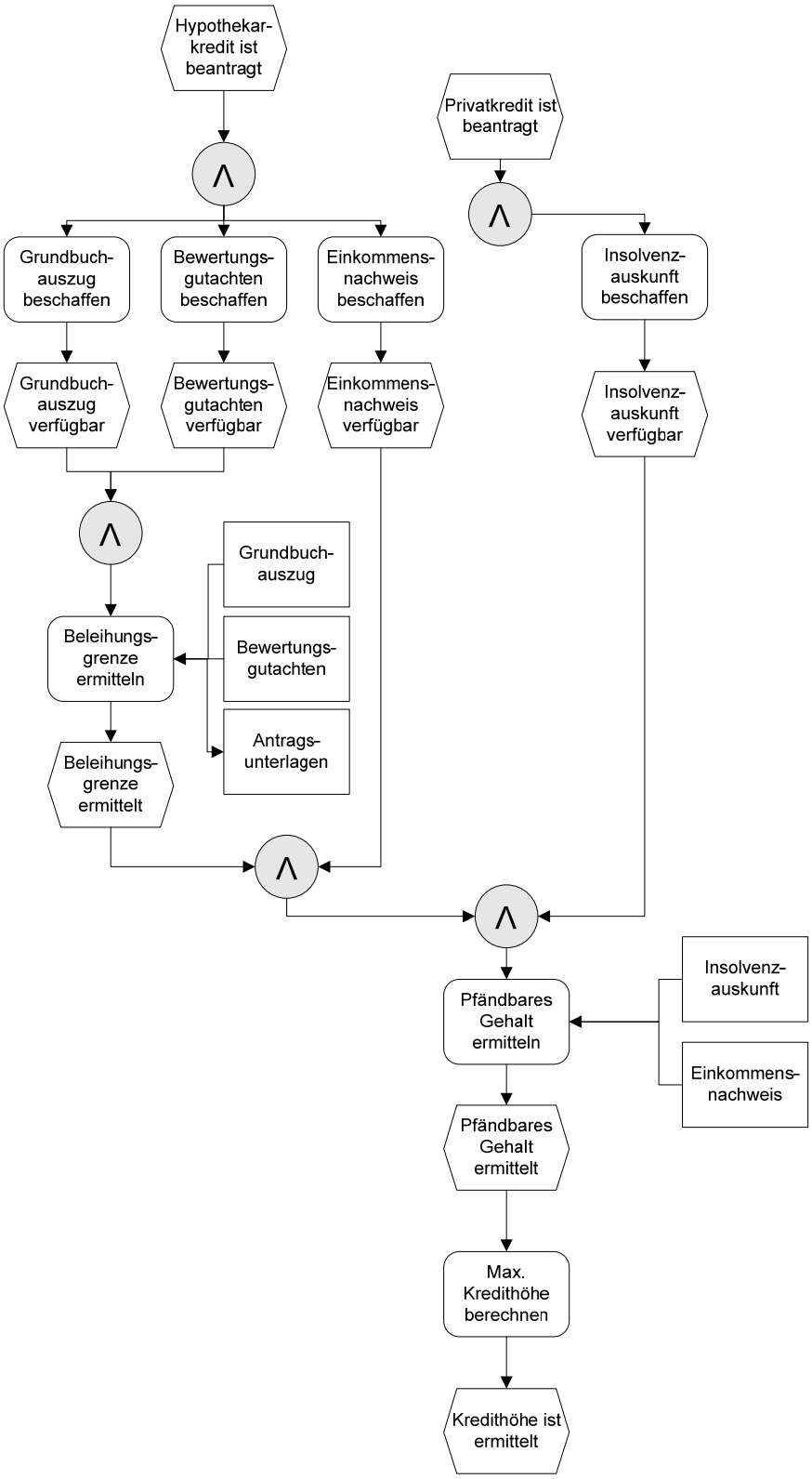



**Abbildung 141: Angepasster Teilentwurf**

*Aus dem Prozessmodell model_100 muss folgender Teilentwurf in den Solution-Teil kopiert werden, da dieser die maximale Überdeckung lieferte:*

$E_1 := \{$ *'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt', 'Kreditvertrag erstellen', 'Kunde informieren', 'Brief', 'Kreditantrag bearbeitet'* $\}$

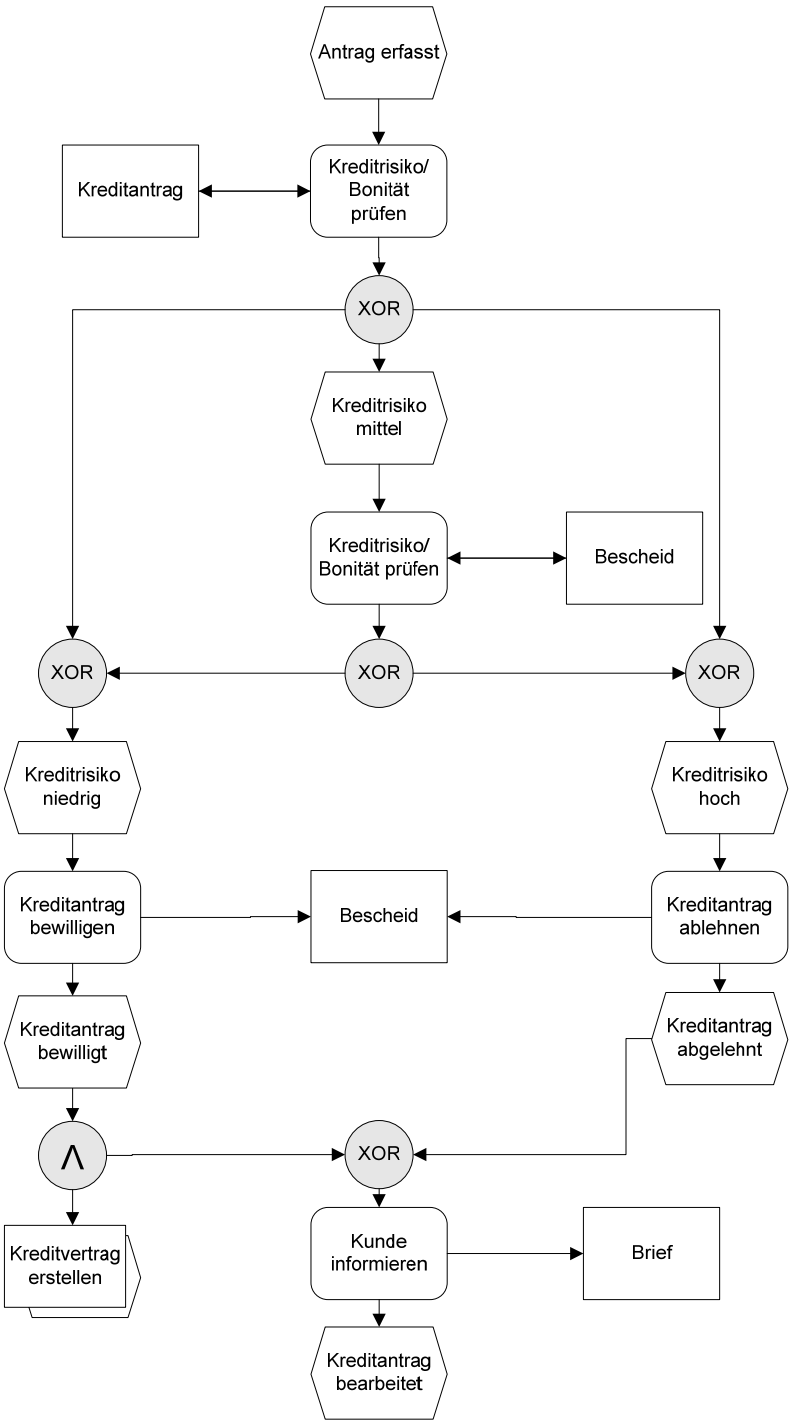



**Abbildung 142: Kopierter Teilentwurf aus Prozessmodell model_100**

*Somit ergibt sich folgende* $c_{query}$*:*

$c_{query} = (Query, Solution = $ *'Hypothekarkredit ist beantragt', 'Grundbuchauszug beschaffen', 'Grundbuchauszug verfügbar', 'Bewertungsgutachten beschaffen', 'Bewertungsgutachten verfügbar', 'Privatkredit ist beantragt', 'Einkommensnachweis verfügbar', 'Einkommensnachweis verfügbar', 'Insolvenzauskunft beschaffen', 'Insolvenzauskunft verfügbar', 'Beleihungsgrenze ermitteln', 'Beleihungsgrenze ermittelt', 'Grundbuchauszug', 'Bewertungsgutachten', 'Antragsunterlagen', 'Antrag erfasst', 'Kreditrisiko/Bonität prüfen', 'Kreditantrag', 'Kreditrisiko mittel', 'Kreditrisiko hoch', 'Kreditantrag ablehnen', 'Bescheid', 'Kreditantrag abgelehnt', 'Kreditrisiko niedrig', 'Kreditantrag bewilligen', 'Kreditantrag bewilligt', 'Kreditvertrag erstellen', 'Kunde informieren', 'Brief', 'Kreditantrag bearbeitet')*

**Entwurfsschemata konfigurieren (Schritt 4):** Gemäß dem Ansatz nach Krampe können auch Entwurfsschemata konfiguriert werden, wenn in der Retrieve-Phase Entwurfselemente eines Referenzmodells ausgewählt wurden. Entwurfsschemata sind Elemente eines Referenzmodells, die konfigurierbar sind und Prozesslösungen für verschiedene Elementvarianten zur Verfügung stellen (z.B. zusätzliche Ablaufschritte in einem Referenzmodell, wenn es sich um einen Konzern handelt). Krampe definiert ein *Entwurfsschema* wie folgt: *„Ein Entwurfsschema ist ein abstrahiertes, wieder verwendbares Entwurfselement, versehen mit Wissen, wie es im aktuellen Anwendungskontext einzusetzen ist.“* Für eine ausführliche Definition sei hier auf Krampe verwiesen. [Kram98, 71, 105ff] Für die Analyse im Rahmen dieser Diplomarbeit wird auf Entwurfsschemata verzichtet, da das Rahmenbeispiel klein gehalten ist.

**Unbekannte Entwurfselemente einlesen (Schritt 5):** In der Retrieve-Phase können auch Entwurfsmerkmale angegeben werden, die noch in keinem Entwurf bzw. Fallbeispiel enthalten sind, die aber zur Erstellung des neuen Entwurfs notwendig sind. Krampe beschreibt die Vorgehensweise folgendermaßen: *„Dazu hat er lediglich die Position des neuen Entwurfselements innerhalb der Indexbäume festgelegt. Im letzten verbleibenden Schritt der Reuse-Phase wird der Softwareingenieur aufgefordert nähere Angaben zu diesen Entwurfselementen zu machen. Durch die Angabe der Position in den Indexbäumen kann das Werkzeug ihn hierbei unterstützen, indem es vorhandene Entwurfsschemata präsentiert, die für Elemente in der entsprechenden Position des Indexbaums geeignet sind.“* [Kram98, 108]

Für die Untersuchung im Rahmen dieser Diplomarbeit wird auf das Einlesen unbekannter Entwurfselemente ebenfalls verzichtet, da dies für die Untersuchung nicht notwendig ist und die Erstellung von Bäumen nicht in die Untersuchung mit einbezogen wurde.

### 5.3.4 Überprüfung der angepassten Prozesslösungen

Nachdem nun für das neue Problem eine Entwurfslösung während der Reuse-Phase gefunden wurde, wird in der Revise-Phase diese Lösung überprüft. Die Notwendigkeit dieser Phase kann sich beispielsweise durch unvollständige Beschreibung des Entwurfsproblems oder auch durch nicht zufriedenstellende Heuristiken, um zu der Lösung zu gelangen, ergeben. [Kram98, 109f]

Krampe sieht in der Revise-Phase eine interne und externe Prüfung vor. Während der *internen Prüfung* werden die Inkonsistenzen der Modelle und Domänen durch die bereits erwähnten Metamodellkonsistenz- und Referenzmodellregeln aufgedeckt. Dazu existieren für die

Behebung automatisierbare Strategien. Bei einer Verletzung der Konsistenzregel wird das die Regel verletzende Element durch ein – falls vorhanden – ähnliches Element ausgetauscht, das nicht gegen die Konsistenzregel verstößt. Hat jedoch der Benutzer das Element ausgewählt, bleibt dieses Entwurfselement aktiviert und die Beziehungen zu den nächstgelegenen Elementen werden solange geändert *„bis die Regel erfüllt ist“*. Wenn diese Konsistenzregel jedoch nie erfüllt werden würde, muss der Benutzer den Fehler beheben, der entsprechend markiert wurde. Die *externe Prüfung* bedeutet, dass sich die Entwurfslösung *„in der realen Welt“* bewähren muss. Die gegebenenfalls auftretenden, notwendigen Änderungen werden hier behoben. [Kram98, 110ff]

***Beispiel 5.3.4.1***

*Auf eine interne Prüfung wird im Rahmen der Diplomarbeit nicht eingegangen. Durch die externe Prüfung erkennt der Benutzer, dass die Prozessmodelle noch angepasst werden müssen. Beim Prozessmodell aus Abbildung 141 müssen noch Elemente entfernt werden (Ereignis „Privatkredit ist beantragt“, UND-Konnektor und eine Beziehung zwischen der Funktion „Insolvenzauskunft beschaffen“, überflüssige Beziehungen und „Hypothekarkredit ist beantragt“ einfügen). Somit ergibt sich das in Abbildung 143 dargestellte geänderte Prozessmodell.*

*Um eine zeitlich-logische Abfolge zu schaffen, werden das Prozessmodell aus Abbildung 143 und aus Abbildung 142 nacheinander angeordnet. Dabei muss das Startereignis „Antrag erfasst“ gelöscht werden. Somit konnte durch die Anwendung des CBR-Cycles die Aufgabenstellung aus Abschnitt 1.3.1 gelöst werden und ein zu Abschnitt 1.3.2 adäquates Prozessmodell geschaffen werden. Auf eine Darstellung des erzeugten Prozessmodells wird verzichtet und der Interpretation des Lesers überlassen.*

*Auch hier können zur Verbesserung der Übersichtlichkeit Prozessschritte durch Prozesswegweiser verdeckt werden und in einer gesonderten Prozesskette angeführt werden.*

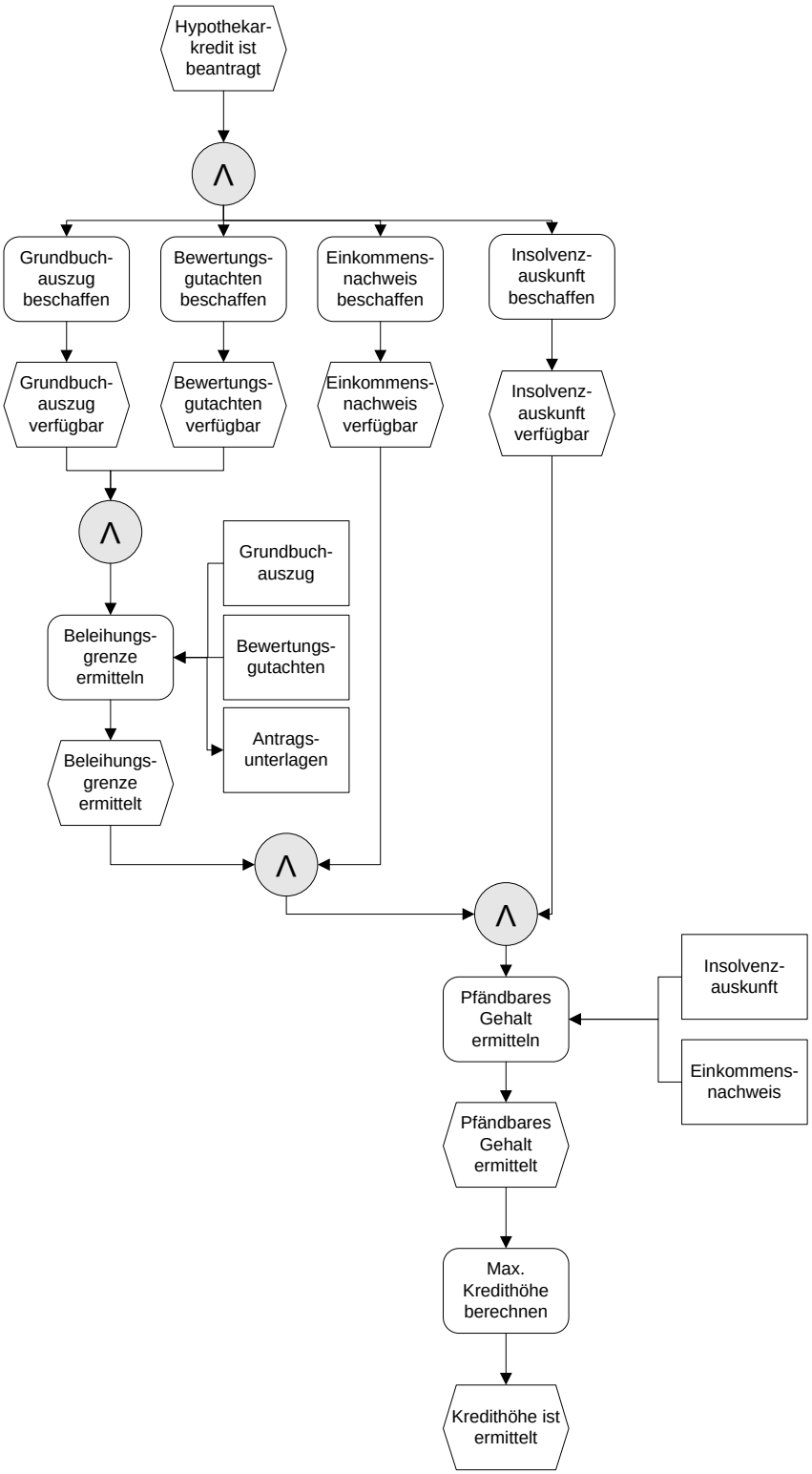



**Abbildung 143: Manuell angepasster Teilentwurf**

### 5.3.5 Integration neuer Prozesslösungen in die Wissensbasis

Diese Phase dient dazu, die neue Entwurfslösung in die Entwurfsbibliothek zu integrieren und das Problemlöseverhalten entsprechend zu verändern. Dabei müssen folgende Aufgaben durchgeführt werden [Kram98, 163ff]:

- Anwendbarkeit der Lösung bestimmen und integrieren
- Anpassung der Wissensbasis

Um die Anwendbarkeit der Lösung und die gemachten Erfahrungen in das soeben erstellte Fallbeispiel zu integrieren, werden folgende Attribute verwendet (vgl. Tabelle 7, Seite 142): *description, context, outcome, alternate_solution, rejected_solution* und *status*. Hier werden die Voraussetzungen notiert, unter denen der Entwurf anzuwenden ist (Anwendungskontext), gemachte Erfahrungen, alternative bzw. abgelehnte Lösungen und den Status angegeben. Der Status kann die Ausprägungen *learned* (gelernter Fall), *work-in-progress* (Fall noch in Bearbeitung) und *has-to-be-justified* (Anwendbarkeit wurde noch nicht bestätigt) haben.

Die Wissensbasis muss durch den Benutzer angepasst werden. Dazu muss er die Konsistenzregeln, eventuell auch Entwurfsschemata und die Indizierung neu hinzugekommener Elemente anpassen. Für die Anpassung der Konsistenzregeln bzw. der Entwurfsschemata (Änderung des *„Methoden- und Domänenwissens“*) ist bei diesem Prototypen eine Änderung des Prolog-Quellcodes notwendig, die durch den Benutzer durchgeführt wird. Durch diesen wird die Wissensbasis aufgebaut, verwaltet und editiert. Die Änderung der Indexkategorien kann durch den Web-Browser realisiert werden.

# 6 Analyse und Vergleich der verwendeten Ansätze

In diesem Abschnitt werden nun die ausgewählten Ansätze der reaktiven Wiederverwendung analysiert und gegenüber gestellt. Das Ziel dieser Untersuchung ist, jenen Ansatz zu identifizieren, der den gesamten Ablauf der reaktiven Wiederverwendung unterstützt. Aus diesem Grund werden Kriterien definiert, die eine Aussage darüber geben, ob und in welchem Ausmaß eine Unterstützung durch die Konzepte gegeben ist. Damit die Stärken und Schwächen der einzelnen Ansätze den jeweiligen Aufgaben der reaktiven Wiederverwendung zugeordnet werden können, werden die Kriterien anhand der gemeinsam verwendeten Konzepte aus Abschnitt 2.2.2.1 zugeordnet.

Reaktive Modellierungsansätze müssen gewisse Anforderungen erfüllen, die es herauszufinden gilt und die für die Evaluierung und Bewertung der einzelnen Ansätze herangezogen werden. Zuerst werden die Anforderungen, die an die Reaktive Wiederverwendung gestellt werden, aus der Literatur (vgl. [BöKS06]) identifiziert.

In Abschnitt 6.1 wird kurz auf die grundlegenden Anforderungen an die reaktive Wiederverwendung eingegangen und Kriterien daraus abgeleitet.

Die Bewertung der Ansätze erfolgt in den darauf folgenden Abschnitten. Dazu werden die Bedeutungen der Evaluationskriterien erklärt. In Abschnitt 6.2 werden die die Konzepte betreffenden Kriterien der einzelnen Ansätze verglichen und anschließend eine Bewertung abgegeben. Dazu werden die Kriterien den gemeinsam verwendeten Konzepten der reaktiven Wiederverwendung zugeordnet und ihre Bedeutung erklärt. In Abschnitt 6.3 wird die Umsetzung der Ansätze verglichen, indem Kriterien zu den jeweiligen Prototypen untersucht werden. Diese Unterteilung zwischen Konzept und Prototyp wird eingeführt, da das Konzept auch durch einen differenten Prototypen realisiert werden kann. Am Ende jeden Abschnitts werden die Bewertungen der einzelnen Ansätze in eine Bewertungsmatrix eingefügt, wobei in den jeweiligen Zellen die Bewertung der einzelnen Ansätze zu dem Kriterium eingetragen wird und so eine Gegenüberstellung der Ansätze erfolgt.

Abschnitt 6.4 fasst die Stärken und Schwächen überblicksmäßig zusammen.

## *6.1 Anforderungen an die reaktive Wiederverwendung*

Grundsätzlich sollen Ansätze der reaktiven Wiederverwendung die Suche und die Auswahl nach geeigneten Artefakten unterstützen. Die Suche soll durch einen strukturierten und eindeutigen Aufbau der Wissensbasis erleichtert werden, wobei beachtet werden soll, dass redundante Objekte nicht mehrfach gespeichert werden sollen [Kram98, 67ff]. Die Auswahl soll einerseits durch Strategien der Priorisierung (z.B. Ähnlichkeitsbestimmung nach Krampe) und andererseits durch Visualisierung der Prozessschritte erleichtert werden. Die Artefakte sollen Prozesslösungen zur Lösung eines bestimmten Problems zur Verfügung stellen. Die Unterstützung der Wiederverwendung kann durch eine größere Auswahlmöglichkeit verbessert werden. Daher sollen auch ähnliche Problemlösungen vorgeschlagen werden. Außerdem gehen manche Autoren davon aus, dass der Anpassungsaufwand geringer ist, als der Aufwand für eine Neuerstellung eines Prozessmodells zur Lösung des Problems (vgl. z.B. [Kram98, 96f]). Dies führt in weiterer Folge zur Notwendigkeit Strategien zur Anpassung zur Verfügung zu stellen, um gefundene, ähnliche Artefakte an die neue Problemsituation durch

das Konzept anpassen zu können und somit den Anpassungsaufwand noch weiter zu minimieren. Um den Grad der Wiederverwendung zu erhöhen, sollen unterschiedliche Abstraktionsgrade der Artefakte zur Verfügung gestellt werden. Die Treffsicherheit wiederverwendbarer Artefakte steigt mit der Anzahl vorhandener Artefakte in der Wissensbasis [Kram98, 36]. Aus diesem Grund ist es von Vorteil, Referenzlösungen, damit sind hier Referenzmodelle gemeint, zur Wiederverwendung mit einzubeziehen. Doch auch spezifischere Unternehmensmodelle können sinnvolle Teilprozesse enthalten, die wiederverwendet werden können. Daher sollen auch diese wiederverwendet werden können. Um diese wieder gemäß der Prozessmodellierung in eine zeitlich-logische Anordnung zu bringen, sollen zwischen den Artefakten Beziehungen definiert werden können, wobei die Beziehungen und auch die Artefakte dabei eindeutig sein sollen und daher formal spezifiziert werden können [Hage05, 5]. Aus diesem Grund soll das Konzept bei der Wiederverwendung eine Unterstützung zur Anpassung an den Anwendungskontext bieten und so die Artefakte in eine zeitlich-logische Anordnung bringen. Nach der Anpassung sollen die Artefakte auf Konsistenz überprüft werden, und zwar sowohl im Hinblick auf die eingesetzte Prozessmodellierungsmethode, als auch auf die vom jeweiligen Konzept definierten Regeln.

Der Benutzer soll nach den Anpassungen und der Überprüfung durch das System ebenfalls die Möglichkeit haben, diese zu überprüfen. Dazu sollen die Anpassungen visualisiert werden. Desweiteren sollen die gemachten Erfahrungen während des Prozesses der reaktiven Modellierung zu den durch die Wiederverwendung erstellten Prozessmodellen dokumentiert werden können, um von „Best Practices“ zu profitieren. Da die erstellten Prozessmodelle ebenfalls für zukünftige Wiederverwendungen nutzbar gemacht werden sollen, ist es von Vorteil, wenn das Konzept die Einordnung in die Wissensbasis unterstützt. Die Konzepte sollen eindeutig verständlich sein. Daher sollen die verwendeten Begriffe definiert werden [Hage05, 5].

Die Forderung nach Mehrbenutzerfähigkeit bedingt unter anderem auch Anforderungen wie Ort- und Plattformunabhängigkeit oder auch die Verwendung des Werkzeugs als Dokumentationswerkzeug, die an die prototypische Umsetzung des Konzepts zu stellen sind.

Wie aus dem Abschnitt 2.1 erkennbar ist, wird die Prozessmodellierung in unterschiedlichen Klassen von Unternehmen angewendet, wobei die Prozessmodelle in verschiedenen Prozessmodellierungsmethoden erstellt werden. Daher ergibt sich die Notwendigkeit, dass Ansätze der reaktiven Wiederverwendung einerseits methodenunabhängig und andererseits domänenunabhängig sind, um sie in vielen Bereichen einsetzen zu können.

Aus diesen genannten Anforderungen können nun folgende Kriterien abgeleitet werden, die einerseits dem Konzept und andererseits dem Prototypen zugeordnet werden können. Die Kriterien zum Konzept können weiter in die in Abschnitt 2.2.2.1 eingeführte Einteilung unterteilt werden.

- Allgemein
  - Begriffsdefinition
  - Methodenunabhängigkeit
  - Domänenunabhängigkeit
- Integration von externen Prozesslösungen
  - Strukturierter Aufbau der Wissensbasis
  - Unterschiedliche Abstraktionsstufen
  - Berücksichtigung von Referenzlösungen
  - Atomare Zerlegung eines Prozessmodells

    - Einheitliche Beschreibung der wieder zu verwendenden Artefakte
    - Beschreibung der Beziehungen zwischen den wieder zu verwendenden Artefakten
    - Formale Beschreibung der Prozesslösungen der wieder zu verwendenden Artefakte
    - Grad der automatisierten Unterstützung beim Befüllen der Wissensbasis
- Suche nach geeigneten Prozesslösungen
    - Grad der Unterstützung bei der Suche nach Artefakten
    - Grad der Unterstützung bei der Auswahl der Artefakte
    - Interaktion mit Benutzer
- Anwendung gefundener Prozesslösungen
    - Grad der Unterstützung bei der Wiederverwendung der gefundenen Prozesslösungen
    - Grad der Unterstützung bei der kontrollierten Anpassung an den Anwendungskontext
    - Interaktion mit Benutzer
    - Flexibilität bei Änderung des Anwendungskontexts
    - Grad der Unterstützung beim Hinzufügen weiterer Details
- Überprüfung angepasster Prozesslösungen
    - Konsistenzprüfung durch das System
    - Überprüfung der angepassten Prozesslösung durch den Benutzer
- Integration neuer Prozesslösungen in die Wissensbasis
    - Dokumentation von erprobtem Wissen
    - Grad der Unterstützung bei der Einordnung des wieder zu verwendenden Artefakts in die Wissensbasis

Dem Prototyp können folgende Kriterien zugeordnet werden:

- Schnittstellen zu anderen Modellierungswerkzeugen
- Visualisierung der vorgeschlagenen Artefakte zur Entscheidungsunterstützung
- Visualisierung der angepassten Artefakte
- Verwendung als Dokumentationswerkzeug
- Ort- und Plattformunabhängigkeit

Zu jedem Kriterium wird im folgenden Abschnitt untersucht, inwieweit dieses von dem jeweiligen Ansatz erfüllt wird.[21] Wird ein Kriterium vollständig erfüllt, wird es mit **hoch** bewertet, wird es teilweise erfüllt, wird es mit **mittel** bewertet. Wird ein Kriterium jedoch nicht unterstützt, wird es mit **niedrig** bewertet. Abschließend werden die entsprechenden Ausprägungen in die Bewertungsmatrix eingetragen.

## *6.2 Konzeptionelle Gegenüberstellung*

Diese Kriterien betreffen das Konzept der jeweiligen Ansätze. Sie werden in die in Abschnitt 2.2.2.1 eingeführten gemeinsam verwendeten Konzepte eingeteilt. Die allgemeinen Kriterien zu Beginn können den Konzepten nicht zugeordnet werden und werden daher separat angeführt. Zu jedem Ansatz wird eine Bewertung abgegeben.

[21] Die Kriterien werden in Anlehnung an die Anforderungen an die reaktive Wiederverwendung aus [Hage05, Rupp02, Kram98, BöKS06] abgeleitet.

***Kriterium: Begriffsdefinition***

Um einer unterschiedlichen Interpretation und Nutzung der Artefakte entgegenzuwirken, müssen die benötigten Elemente definiert werden. Ein allgemeines, jedoch nicht minder wichtiges Kriterium ist daher eine einheitliche Begriffsdefinition, wie die Elemente der Artefakte zu deuten und zu nutzen sind. [Hage05, 5]

*Ansatz nach Hagen:*

Hagen beschreibt alle für ihr Konzept notwendigen Begriffe (u.a. Prozessmuster, Domäne, Prozessmusterkatalog) [Hage05, 45ff]. Durch eine einheitliche Begriffsdefinition werden Unklarheiten über die Interpretation beseitigt.

Bewertung: Hoch

*Ansatz nach Rupprecht:*

In diesem Ansatz gibt es keine einheitliche Begriffsdefinition, z.B. Abhängigkeitstypen der Gestaltungsregeln werden auch Aktionstypen bzw. Beziehungstypen genannt. [Rupp02, 90ff]

Bewertung: Niedrig

*Ansatz nach Krampe:*

Die wesentlichen Begriffe wurden für diesen Ansatz geklärt [Kram98, 73ff]. Daher ist die zum Verständnis erforderliche Begriffsdefinition erfüllt

Bewertung: Hoch

***Kriterium: Methodenunabhängigkeit***

Methodenunabhängigkeit als Kriterium bedeutet, dass Prozesslösungen in die Wissensbasis aufgenommen werden können, unabhängig von ihrer Modellierungsmethode, mit der sie erstellt wurden.

*Ansatz nach Hagen:*

Dieser Ansatz ist eine Erweiterung des UML-Aktivitätsdiagramms und führt eine Erweiterung der formalen Syntax und Semantik für Prozessmuster ein [Hage05, 6].

Daher ist die Methodenunabhängigkeit nicht gegeben.

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Dieser Ansatz wurde *„weitgehend unabhängig von der eingesetzten Prozessmodellierungsmethode und -werkzeugunterstützung“* erstellt [Rupp02, 109]. Es wurde ein *„prototypischer Entwurf der Benutzeroberfläche“* mit Hilfe des Prozessmodellierungswerkzeugs POWM (Prozessorientiertes Wissensmanagement) erstellt. Dies ist jedoch nur eine Möglichkeit dieses Konzept zu realisieren. [Rupp02, 126] Die Funktionalitäten, die POWM zur Verfügung stellt, wurden in diesem Ansatz nicht berücksichtigt (z.B. Benutzerverwaltung und Basisfunktionalitäten [Rupp02, 8] wie verteilte Prozessmodellierung, Konsistenzhaltung zwischen Prozesstypen und ihren Instanzen und die Möglichkeit Prozesse hierarchisch zu strukturieren [Rupp02, 130]).

Aufgrund dieser Einschränkungen wird dieses Kriterium nur teilweise erfüllt.

Bewertung: Mittel

*Ansatz nach Krampe:*

Obwohl Krampe in seiner Dissertation das Entity-Relationship-Modell, die Funktionsbäume und die ereignisgesteuerte Prozesskette verwendet hat, ist dieser Ansatz methodenunabhängig. Es ist lediglich Bedingung, dass die verwendete Modellierungsmethode ein Metamodell, Metakomponenten, Beziehungen untereinander und Konsistenzregeln besitzt. Dabei ist eine Änderung der Unterscheidungsmerkmale notwendig, die auf die verwendeten Methoden abgestimmt sind. Der Zugriff auf die einzelnen Entwurfselemente über die Kategorien bleibt gleich. [Kram98, 116f]

Bewertung: Hoch

***Kriterium: Domänenunabhängigkeit***

Durch dieses Kriterium soll gewährleistet werden, dass der Ansatz nicht nur auf eine bestimmte Anwendungsdomäne ausgerichtet ist, um dieses Werkzeug auch für andere Anwendungsdomänen verwenden zu können.

*Ansatz nach Hagen:*

Der Ansatz ist zwar auf die Domäne der Softwareentwicklung ausgerichtet [Hage05, 46]. Jedoch können abhängig von den verwendeten Prozessmodellen auch Prozessmuster aus unterschiedlichen Domänen dokumentiert werden und strukturiert in einem Prozessmusterkatalog repräsentiert werden [Hage05, 20]. Dazu müsste die Syntax des Prozessmusters lediglich um ein weiteres Attribut zur Angabe der verwendeten Domäne erweitert werden [Hage05, 73f].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Die Domäne, die bei diesem Ansatz verfolgt wird, ist die *„Anwendungsdomäne projekthafter Prozesse“* [Rupp02, 3]. Projekthafte Prozesse sind durch ihre Einmaligkeit und Komplexität gekennzeichnet, die in unterschiedlichsten Anwendungsdomänen angesiedelt sein können [ScSt98, 229].

Obwohl projekthafte Prozesse die Anwendungsdomäne sind, ist hier eine Domänenunabhängigkeit gegeben.

Bewertung: Hoch

*Ansatz nach Krampe:*

Dieser Ansatz ist auf jede beliebige Domäne anwendbar, die über ein Referenzmodell verfügt. Er kann erst dann effizient genutzt werden, sobald er auf eine Domäne ausgerichtet ist. Jedoch können auch domänenübergreifend Fallbeispiele in der Wissensbasis vorhanden sein und wiederverwendet werden. [Kram98, 116f]

Der ist Wiederverwendungsgrad höher, wenn die Fallbeispiele aus derselben Anwendungsdomäne kommen bzw. der CBR-Cycle öfter durchlaufen werden musste, um auf genügend vorhandenes Wissen und damit Fallbeispiele zurückgreifen zu können.

Bewertung: Mittel

### 6.2.1 Integration von externen Prozesslösungen

Mit diesen Kriterien soll bewertet werden, wie hoch der Grad der Unterstützung bei der Integration von externen Prozesslösungen bei erstmaliger Anwendung des Ansatzes ist, wenn die Wissensbasis noch leer ist.

***Kriterium: Strukturierter Aufbau der Wissensbasis***

Dieses Kriterium bewertet, ob die Wissensbasis strukturiert und einheitlich aufgebaut ist. Damit ist gemeint, dass die Zuordnung der Artefakte zu Kategorien eindeutig ist, und sie nicht redundant in mehreren Kategorien gespeichert werden. Sind sie in mehreren Kategorien enthalten, soll nicht eine Kopie, sondern eine Referenz gespeichert werden, um die Artefakte konsistent zu halten.

*Ansatz nach Hagen:*

Durch die formale Definition der Prozessmuster sowie ihrer Beziehungen werden die Prozessmuster in eine eindeutige Anordnung gebracht und können über die Prozessmusterbeziehungen wiedergefunden werden (Top-Down- oder From-Within-Strategie) [Hage05, 169f]. Mit Hilfe von OCL-Regeln wird sichergestellt, dass gespeicherte Prozessmuster nicht ident sind [Hage05, 74].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Die generischen Rahmenbedingungen und Prozessbausteine werden jeweils in einem Ordnungsraster verwaltet, welches durch eine Ordnerstruktur abgebildet ist [Rupp02, 72f]. Für die Ordnerstruktur der generischen Rahmenbedingungen schlägt Rupprecht eine Gliederung vor, die für die Einflussgrößenkategorien auf den oberen zwei Ebenen verwendet und erweitert werden kann, und auf den darunter liegenden Ebenen durch die Benutzer frei gestaltet werden kann [Rupp02, 36ff]. Das Ordnungsraster der Prozessbausteine bzw. der Musterprozesse ist von der Anwendungsdomäne abhängig und kann – analog zu den generischen Rahmenbedingungen – ebenfalls in der zweiten Ebene erweitert und in der unteren Hierarchie selbst gestaltet werden. Prozessbausteine können entweder einer Prozesskategorie zugeordnet oder mit einer Generalisierungs-/Spezialisierungsbeziehung mit einem abstrakteren Prozessbaustein verknüpft werden. Die Prozessbausteine werden jedoch *„nicht konsistent gehalten“* (z.B. bei Änderungen), noch werden Strukturen bzw. Eigenschaften vererbt. [Rupp02, 85] Sowohl Prozessbausteine als auch Rahmenbedingungen können in mehreren Ordnern gespeichert werden [Rupp02, 95, 98].

Die Kategorisierung wird weitgehend den Benutzern überlassen. Auf den oberen Ebenen ist eine Struktur vorgegeben. Jedoch Prozessbausteine und auch Rahmenbedingungen werden in mehreren Kategorien abgelegt. Dieses Kriterium ist damit teilweise erfüllt.

Bewertung: Mittel

*Ansatz nach Krampe:*

Die Kategorisierung erfolgt durch Indexbäume, die das Suchen durch Traversieren der Indexbäume erleichtert. Es wird zwischen allgemeinen und speziellen Unterscheidungsmerkmalen unterschieden. Zu den allgemeinen Unterscheidungsmerkmalen zählen Unterscheidungen nach Funktionsbereich und Abstraktionsgrad. Der Funktionsbereich wird durch das importierte Referenzmodell geprägt. Der Abstraktionsgrad gibt an, ob es sich um ein Referenzmodell oder bereits um ein spezielleres Unternehmensmodell handelt. Bei den speziellen Unterscheidungsmerkmalen wird zwischen Sicht (Datensicht, Funktionssicht und Prozesssicht), Datenart (Stammdaten und Bewegungsdaten) und Datenzugriffsart unterschieden. Die speziellen Unterscheidungsmerkmale sind durch die verwendeten Modellierungsmethoden geprägt. Werden diese Methoden durch andere ersetzt, muss eine neue Kategorisierung vorgenommen werden, beispielsweise nach den Komponenten, die von der Methode verwendet werden. [Kram98, 77ff]

Durch die eindeutigen Vorgaben dieses Ansatzes wird eine Kategorisierung der verwendeten Modelle und Komponenten gut unterstützt.

Bewertung: Hoch

***Kriterium: Unterschiedliche Abstraktionsstufen***

Je abstrakter ein wieder zu verwendendes Artefakt ist, desto größer ist die Wahrscheinlichkeit der Wiederverwendung (vgl. z.B. [Rupp02, 80f]).

*Ansatz nach Hagen:*

Durch die Prozessmusterbeziehungen können die Abstraktionsstufen bestimmt werden. Bei Sequence- oder Processvariance-Beziehung befinden sich die Prozessmuster auf einer ähnlichen Abstraktionsstufe. Durch die Use- und Refinement-Beziehung werden untergeordnete Abstraktionsstufen angegeben. [Hage05, 48]

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Rupprecht unterscheidet zwischen inhaltlicher Abstraktion der Prozessbausteine und *„Abstraktion der Modellierung“*, wobei er hiermit den *„Gestaltungsrahmen für konkretere Modelle“* meint (Metaebene, Typebene, Ausprägungsebene). Bei der Definition von Prozessbausteinen ist es nicht immer erforderlich, von gewissen Abläufen zu abstrahieren, da diese ebenfalls einen hohen Grad der Wiederverwendung aufweisen können. [Rupp02, 48ff] Es gibt jedoch die Möglichkeit abstrakte und spezielle Prozessbausteine zu definieren und über Generalisierungs- und Spezialisierungsbeziehung in der Ordnerstruktur zu verknüpfen [Rupp02, 87].

Bewertung: Hoch

*Ansatz nach Krampe:*

Wie bereits erwähnt stellt das Referenzmodell im Gegensatz zu einem Unternehmensmodell eine abstraktere Sichtweise dar. Desweiteren können Unternehmensmodelle voneinander abgeleitet werden und weiter durch Hinzufügen von Details konkretisiert werden. Dadurch entsteht eine Vererbungsbeziehung zwischen den Prozessmodellen bzw. Komponenten. [Kram98, 36ff]

Somit entsteht eine Hierarchie, welche in der Theorie nicht begrenzt ist.

Bewertung: Hoch

***Kriterium: Berücksichtigung von Referenzlösungen***

Referenzmodelle geben bereits eine Struktur vor. Sie können im weiteren Verlauf des Prozesses der reaktiven Wiederverwendung mit weiteren Details versehen werden. Referenzmodelle dienen als Vorschlag für eine bewährte Lösung und sind auf einem hohen Abstraktionsniveau angesiedelt. [Kram98, 34ff]

*Ansatz nach Hagen:*

Ein Referenzmodellimport ist in diesem Ansatz nicht vorgesehen [Hage05].

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Es ist kein Import für generische Referenzmodelle vorgesehen, da der Schwerpunkt auf dem Herunterbrechen der *„Kausalzusammenhänge zwischen Kontext und Prozess-Strukturen auf einzelne Abhängigkeiten zwischen Rahmenbedingungen und Prozessbausteinen“* liegt. Zusätzlich ist in den meisten Fällen *„eine Adaptierung des Referenzmodells“* notwendig, um sie an den gegebenen Anwendungskontext anzupassen, was zu einem signifikanten Aufwand führen kann und damit *„den hohen potenziellen Nutzen einer (seltenen) Wiederverwendung übersteigen kann“*. [Rupp02, 58]

Bewertung: Niedrig

*Ansatz nach Krampe:*

Es gibt eine Schnittstelle zwischen CBModeler und ARIS-Toolset, das wegen der Ausrichtung auf unterschiedlichste Domänen eine höhere Anzahl an Referenzmodellen zur Verfügung stellt. Diese können in das Repository des CBModelers importiert werden und geben somit eine Struktur der Anwendungsdomäne vor. [Kram98, 132ff]

Bewertung: Hoch

***Kriterium: Atomare Zerlegung eines Prozessmodells***

Um auch flexible Anforderungen bzw. kleinere Projekte unterstützen zu können, muss ein vorhandenes Prozessmodell in weitere Teilprozesse unterteilt werden können. Dieses Kriterium soll eine Aussage darüber geben.

*Ansatz nach Hagen:*

Prozessmuster stellen eine Lösung für *„ein bestimmtes Problem“* zur Verfügung und beinhalten Prozessschritte einzelner Prozessmodelle. Ein Prozessmodell kann in mehrere Prozessmuster aufgeteilt werden, wobei das gesamte Prozessmodell ebenfalls ein Prozessmuster ergibt. Das Prozessmuster, das das gesamte Prozessmodell darstellt, ist ein Kompositmuster und mit Use-Beziehungen mit den Komponentenmustern verbunden. [Hage05, 56]

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Mit Hilfe der Prozessbausteine können *„kleinere, zeitlich und logisch in sich abgeschlossene Einheiten“* aus den Prozessmodellen herausgelöst werden *„und als Prozessbausteine gespeichert werden“*. [Rupp02, 5]

Bewertung: Hoch

*Ansatz nach Krampe:*

Dieser Ansatz wurde mit der Zielsetzung entwickelt, konzeptionelle Entwürfe für Informationssysteme zu erleichtern. Er wird als wenig flexibel eingestuft, da der CBR-Cycle bei Änderungen erneut durchlaufen werden muss. Jedoch können aus Fallbeispielen, die aus verschiedenen Modellen bestehen, einzelne Teile mit Hilfe der Metamodellkonsistenzregeln herausgelöst werden und in einen Teilentwurf übernommen werden (z.B. aus einem EPK-Modell wird nur ein kleiner Abschnitt benötigt). [Kram98, 117ff]

Bewertung: Hoch

***Kriterium: Einheitliche Beschreibung der wieder zu verwendenden Artefakte***

Ein wichtiges Kriterium ist das Vorhandensein eines einheitlichen Beschreibungsschemas der Artefakte und aller notwendigen Komponenten, um Missverständnisse bezüglich Interpretation einiger Elemente bzw. Attribute der Artefakte aus dem Weg zu gehen [Hage05, 3].

*Ansatz nach Hagen:*

Hagen definiert in ihrer Arbeit ein Prozessmusterschema und ein Problemschema, mit denen die Prozessmuster und Probleme einheitlich beschrieben werden können. Dadurch wird einer falschen Interpretation der verwendeten Elemente entgegen gewirkt. [Hage05, 65ff]

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Es wurde kein einheitliches Beschreibungsschema für Prozessbausteine definiert [Rupp02].

Bewertung: Niedrig

*Ansatz nach Krampe:*

Krampe definiert ein Schema für Fallbeispiele (vgl. Tabelle 7), das mit einheitlichen Attributen beschrieben wird. [Kram98, 138f]

Bewertung: Hoch

***Kriterium: Beschreibung der Beziehungen zwischen den wieder zu verwendenden Artefakten***

Das Kriterium der vollständigen Beschreibung des wieder zu verwendenden Artefakts meint die Beschreibung der Beziehungen zwischen den Artefakten. Es ist eine formale Definition der Beziehungen notwendig, um Unklarheiten bezüglich der zeitlich-logischen Anordnung der Artefakte zu beseitigen (z.B. Sequenz oder Parallelität) [Hage05, 3].

*Ansatz nach Hagen:*

Die Prozessmuster können durch die Prozessmusterbeziehungen in eine zeitlich-logische Abfolge gebracht werden. Mögliche Beziehungen sind Sequence-, Use-, Refinement- und Processvariance-Beziehungen. Durch diese Beziehungen ist die zeitlich-logische Abfolge der Prozessmuster klar definiert und es kann zu keinen Missverständnissen kommen. [Hage05, 54ff]

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Die Anordnungsbeziehungen zwischen den wieder zu verwendenden Artefakten (Prozessbausteinen) werden mit Einbauregeln beschrieben. Einbauregeln sind – wie die Kontextgestaltungs- und Prozessanpassungsregeln – Teil der Gestaltungsregeln. Sie *„verknüpfen Prozessbausteine untereinander und speichern Wissen über deren zeitlich-logische Anordnung“*. [Rupp02, 75] Sie geben also Auskunft darüber, ob ein Prozessbaustein *„vor, nach oder parallel zu“* anderen Prozessbausteinen angeordnet werden soll. Gültige Beziehungstypen sind *„ist Vorgänger von“*, *„ist Nachfolger von“*, *„ist parallel zu“* und *„ist zugehörig zu“*. Der letzte Beziehungstyp bezeichnet *„jede andere beliebige Beziehung zwischen zwei Prozessbausteinen“*, die keine allgemeingültige Anordnungsbeziehung darstellen soll. Die Beziehungstypen sind auch umkehrbar, sie gelten daher auch in die andere Richtung. [Rupp02, 89]

Das Kriterium der vollständigen Beschreibung der wieder zu verwendenden Artefakte ist somit erfüllt.

Bewertung: Hoch

*Ansatz nach Krampe:*

Zwischen den Elementen (Komponenten und Modellen), die als Bäume dargestellt werden, gibt es als gültige Beziehungen *is-a-* und *is-modified-Beziehungen*. Die Indexbäume, die dem Zugriff auf Elemente dienen, gibt es eine *is-a-Beziehung*. [Kram98, 85ff] Diese Beziehungen geben noch nicht über den Ablauf der Geschäftsprozesse Auskunft. Dieser ist bei der ereignisgesteuerten Prozesskette beschrieben und wird übernommen, indem die verwendeten Teilentwürfe in die Lösung übernommen werden [Kram98, 105ff].

Bewertung: Hoch

***Kriterium: Formale Beschreibung der Prozesslösungen der wieder zu verwendenden Artefakte***

Es ist notwendig, dass die Prozesslösungen, die den Artefakten zugeordnet werden, formal beschrieben werden. Dies bedeutet, dass es möglich sein soll, die Abfolge von Aktivitäten den Artefakten zuordnen zu können und diese formal definiert sind. [Hage05, 3]

*Ansatz nach Hagen:*

Durch die Verwendung der UML-Aktivitätsdiagramme ist bereits eine formale Syntax für die Darstellung der einzelnen Prozessschritte vorhanden [Hage05, 30]. Hagen erweitert die formale Syntax der UML-Aktivitätsdiagramme um die Konzepte ihres Ansatzes. Zu den Konzepten zählen die Notationen des Prozessmusters, des Problems, des Kontexts, des Prozesses, der Prozessmusterbeziehungen und der Aktivitäten, Objekte, Ereignisse, Rolle und Werkzeug. [Hage05, 69ff] Für diese Konzepte, und auch für die UML-Aktivitätsdiagramme

wird eine formale Semantik definiert, die die informale Beschreibung der formalen Syntax ersetzt [Hage05, 99ff]. Die formale Semantik der UML-Aktivitätsdiagramme wird durch Bedingungs-Ereignis-Systeme realisiert [Hage05, 115].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Wie bereits erwähnt wurde dieser Ansatz *„weitgehend unabhängig von der eingesetzten Prozessmodellierungsmethode und -werkzeugunterstützung“* erstellt. Somit können auch andere Prozessmodellierungsmethoden verwendet werden. [Rupp02, 109] Die Beziehungen zwischen den wieder zu verwendenden Artefakten sind formal definiert, da es keine Doppeldeutigkeiten über ihre Anordnungsbeziehungen gibt (vgl. Beziehungstypen bzw. Aktionstypen der Einbauregeln) [Rupp02, 89].

Dieses Kriterium ist somit erfüllt, sofern Prozessmodellierungsmethoden verwendet werden, die zumindest eine formale Syntax der Prozesse unterstützen und so Anordnungsbeziehungen zwischen Aktivitäten gesetzt werden können.

Bewertung: Hoch

*Ansatz nach Krampe:*

Die Prozesse werden, wie bereits erwähnt, mit Hilfe der ereignisgesteuerten Prozesskette beschrieben. Jedoch können auch andere Modellierungsmethoden zur Darstellung von Prozessen verwendet werden. Durch Metamodellkonsistenzregeln wird eine Konsistenz des Prozessmodells gewährleistet.

Bewertung: Hoch

***Kriterium: Grad der automatisierten Unterstützung beim Befüllen der Wissensbasis***

Mit diesem Kriterium wird untersucht, wie hoch der Grad der Unterstützung durch den jeweiligen Ansatz bei dem Wissenserwerb und damit bei der Dokumentation von Artefakten aus bestehenden Prozessmodellen oder Referenzmodellen ist (Dokumentation von Wissen). Dies ist wichtig, um Aussagen darüber treffen zu können, ob dem Benutzer bei der Einordnung in eine Kategorie und bei der Anordnung der Artefakte Hilfestellung gegeben wird oder ob der Benutzer die Einordnung selbst übernehmen muss.

*Ansatz nach Hagen:*

Für die Dokumentation eines Prozessmusters führt Hagen einheitliche Prozessmusterschema und Problemschema ein [Hage05, 65ff]. Durch die definierten Beziehungen zwischen den Prozessmustern können sie eindeutig angeordnet werden, jedoch muss der Benutzer die Beziehungen manuell pro Prozessmuster festlegen [Hage05, 190]. Durch die Prozessmusterbeziehungen werden die Prozessmuster eindeutig angeordnet und in dem Prozessmusterkatalog aufgenommen (vgl. Abschnitt 3.4.1). Es ist dabei unerheblich, ob von Referenzmodellen oder Prozessmodellen Prozessmuster in die Wissensbasis aufgenommen werden. Die Vorgehensweise bleibt gleich.

Da durch die einheitlichen Schemata und die Prozessmusterbeziehungen die Dokumentation der Prozessmuster unterstützt wird, jedoch die Beziehungen zwischen den Prozessmustern manuell eingefügt werden müssen, ist dieses Kriterium teilweise erfüllt.

Bewertung: Mittel

*Ansatz nach Rupprecht:*

Mit Hilfe der Wissenserwerbskomponente wird das Expertenwissen *„in Form von Prozessbausteinen, Musterprozessen, Rahmenbedingungen und Gestaltungsregeln“* erfasst. Dabei werden Funktionen verwendet: „Prozessbaustein definieren“, „Musterprozess definieren“, „generische Rahmenbedingung definieren“, „Kontextgestaltungsregel definieren“, „Prozessanpassungsregel definieren“, „Einbauregel definieren“. [Rupp02, 92ff]

Mit Hilfe der Funktionen können die Prozessbausteine manuell in die Wissensbasis aufgenommen werden. Auch bei diesem Ansatz können Prozessbausteine sowohl von Referenzmodellen als auch von Prozessmodellen definiert werden. Der Benutzer wird nicht weiter unterstützt. Dieses Kriterium ist teilweise erfüllt.

Bewertung: Mittel

*Ansatz nach Krampe:*

Die Wissensbasis kann durch ein abgeschlossenes Fallbeispiel in der Retain-Phase erweitert [Kram98, 112]. Je öfter der CBR-Cycle durchlaufen wird, desto größer ist der Umfang der Wissensbasis, auf die zurückgegriffen werden kann [Kram98, 150ff].

Ohne den CBR-Cycle zu durchlaufen, können somit keine Prozessmodelle dokumentiert werden.

Bewertung: Niedrig

### 6.2.2 Suche nach geeigneten Prozesslösungen

***Kriterium: Grad der Unterstützung bei der Suche nach Artefakten***

Dieses Kriterium soll eine Aussage darüber geben, inwiefern der Benutzer bei der Suche in der Wissensbasis unterstützt wird. Es ist ein wesentlicher Punkt für die Wiederverwendung in Frage kommende Artefakte zu suchen und dabei von dem jeweiligen Ansatz gute Unterstützung zu erhalten.

*Ansatz nach Hagen:*

Die Suche erfolgt über die *„Identifikation eines Problems“*, indem der initiale und der resultierende Kontext festgelegt werden [Hage05, 205]. Der Kontext setzt sich aus Objekten sowie Ereignissen zusammen [Hage05, 51]. Über die Verknüpfung zwischen Problem und Prozessmuster können die lösenden Prozessmuster durch diese Vorgehensweise gefunden werden (vgl. Abschnitt 3.4.2).

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Generische Prozessmodelle können durch freies Suchen (Browsen) in der Ordnerstruktur der generischen Prozessmodelle, aber auch zielgerichtet gesucht werden [Rupp02, 102]. Eine Unterstützung bei der Suche erhält der Benutzer, indem er Projektrahmenbedingungen festlegt. Durch die definierten Gestaltungsregeln (Kontextanpassungs-, Prozessanpassungs- und Einbauregel) können weitere Rahmenbedingungen vorgeschlagen werden, und in weiterer Folge die benötigten Prozessbausteine gefunden werden. [Rupp02, 124] Der

Benutzer findet durch das Festlegen der Projektrahmenbedingungen die benötigten Prozessbausteine (vgl. Abschnitt 4.3.2).

Bewertung: Hoch

*Ansatz nach Krampe:*

In diesem Ansatz werden einfache Suchstrategien und Strategien der Ähnlichkeitsbestimmung verwendet. Der Benutzer legt die gewünschten Entwurfsmerkmale zu Beginn des CBR-Cycles fest [Kram98, 17]. Anschließend können Fallbeispiele mit folgenden Suchstrategien gesucht werden: die Suche nach Modellen und die Suche nach Komponenten, wobei diese auch kombiniert werden können [Kram98, 77]. Dadurch können einerseits gesamte Modelle und andererseits Teile von Modellen gefunden werden (vgl. Abschnitt 5.3.2).

Bewertung: Hoch

***Kriterium: Grad der Unterstützung bei der Auswahl der Artefakte***

Mit diesem Kriterium wird untersucht, ob der Ansatz Strategien zur Verfügung stellt, damit dem Benutzer die Auswahl der Artefakte erleichtert wird, wie beispielsweise durch Ähnlichkeitsbestimmung und Priorisierung.

*Ansatz nach Hagen:*

Über den initialen und den resultierenden Kontext werden die Probleme identifiziert und die dazugehörigen Prozessmuster vorgeschlagen. Ein Problem kann auch von mehreren Prozessmustern gelöst werden. Diese Prozessmuster stellen Prozessvarianten dar. Durch den Kontext werden somit alle lösenden Prozessmuster identifiziert und der Benutzer kann das geeignetste Prozessmuster auswählen. [Hage05, 205]

Dieser Ansatz liefert dem Benutzer bei Vorhandensein von mehreren Prozesslösungen alle Prozessvarianten. Jedoch muss er diese selbst auswählen und erhält keine Entscheidungsunterstützung.

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Nachdem die Prozessbausteine bzw. Musterprozesse mit Hilfe der Gestaltungsregeln (Kontextgestaltungsregeln, Prozessanpassungsregeln sowie Einbauregeln) aufgedeckt werden, werden sie dem Benutzer vorgeschlagen, der sie annehmen oder ablehnen kann [Rupp02, 117ff]. Bei mehreren Prozessvarianten muss der Benutzer die Auswahl selbst treffen, und es wird keine Priorisierung durchgeführt [Rupp02, 121].

Bewertung: Niedrig

*Ansatz nach Krampe:*

Der Ansatz schlägt am Ende der Retrieve-Phase einige Fallbeispiele vor, die mit Hilfe der Ähnlichkeitsbestimmung priorisiert werden. Dabei wird auf das Kontrollwissen zurückgegriffen. In der Reuse-Phase werden anschließend noch weitere Teilentwürfe bestimmt (logisch zusammengehörige Entwürfe), die der Benutzer in der Retrieve-Phase nicht ausgewählt hat. Dazu kommen Metamodellkonsistenz-, Referenzmodellregeln und semantische Beziehungen zum Einsatz. Im nächsten Schritt werden iterativ jene Teilentwürfe bevorzugt, die die meisten Entwurfsmerkmale beinhalten. [Kram98, 89ff]

Die Gesamtheit dieser Aufgaben wird durch diesen Ansatz unterstützt.

Bewertung: Hoch

***Kriterium: Interaktion mit Benutzer***

Dieses Kriterium ist wichtig, damit der Benutzer die Auswahl beeinflussen kann. Daher soll der Benutzer ein ausgewähltes Artefakt gegebenenfalls auch ablehnen können. Dieses Kriterium bestimmt daher, ob dies möglich ist.

*Ansatz nach Hagen:*

Der Benutzer kann bei mehreren Prozesslösungen die geeignetste Prozesslösung aussuchen und auch ablehnen [Hage05, 205].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

In der Projektinitialphase dieses Ablaufmodells ist mehrmaliges Setzen der Projektrahmenbedingungen notwendig. Nach dem Setzen der ersten *„Projektrahmenbedingungen auf der obersten Ebene der Prozesshierarchie für das Gesamtprojekt“* (Schritt 1), müssen in jeder Ebene der Prozesshierarchie (Schritt 9) bzw. nach dem Verwenden eines Prozessfalles (Schritt 4) bzw. eines Musterprozesses (Schritt 7) weitere Projektrahmenbedingungen gesetzt werden. [Rupp02, 123ff]

Der Benutzer kann während des Projekts des Öfteren den Anwendungskontext anpassen und die Auswahl beeinflussen.

Bewertung: Hoch

*Ansatz nach Krampe:*

Der Benutzer kann die Entwurfsmerkmale auswählen (Indexmerkmale, Komponententypen und/oder Modelltypen). Nachdem der Benutzer die entsprechende Auswahl getätigt hat (z.B. Funktionen) werden die möglichen Funktionen angezeigt. Das Ergebnis der Retrieve-Phase ist eine Auswahl der am geeignetsten erscheinenden Entwürfe. Erst in der Revise-Phase kann der Benutzer Anpassungen vornehmen. [Kram98,89ff]

Bewertung: Niedrig

### 6.2.3 Anwendung von gefundenen Prozesslösungen

***Kriterium: Grad der Unterstützung bei der Wiederverwendung der gefundenen Prozesslösungen***

Dieses Kriterium soll eine Aussage darüber treffen, wie gut dieser Ansatz den Benutzer bei der Wiederverwendung unterstützt und Strategien zur Generierung von Lösungen anbietet.

*Ansatz nach Hagen:*

Nachdem die Prozessmuster zur Lösung für das aktuelle Problem ausgewählt wurden, müssen sie an den neuen Kontext angepasst werden. Da die Prozessmuster von der Realität abstrahieren können, um den Grad der Wiederverwendbarkeit zu erhöhen, müssen sie

instanziiert werden, indem zusätzlich benötigte Details hinzugefügt werden (beispielsweise zusätzliche Prozessschritte, Objekte und/oder Ereignisse). [Hage05, 205]

Der Grad der Unterstützung geht über das Hinzufügen von weiteren Details nicht hinaus und bietet keine Strategien zur Generierung von Lösungen an.

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Die Wiederverwendung besteht bei diesem Ansatz aus den Schritten 10, 11, 14 und 15 des Vorgehensmodells (vgl. Abbildung 88 auf Seite 113 und Abschnitt 4.3.3). Dabei erfolgt bei Schritt 10 eine Anpassung der Prozessmodellinstanzen durch die Funktion „Teilprozess anpassen“ anhand der ausgewählten Projektrahmenbedingungen, wodurch Prozessbausteine zum Einbau vorgeschlagen werden. Anschließend erfolgt eine manuelle Anpassung[22] der Prozessmodellinstanzen durch den Benutzer (Schritt 11). Nach der manuellen Anpassung können wiederum Projektrahmenbedingungen hinzugefügt werden (Schritt 14). Durch das Auswählen von Projektrahmenbedingungen und die anschließende *„Anpassung der Prozessmodellinstanzen“* können Inkonsistenzen zwischen den Prozessmodellinstanzen und den ausgewählten *„Rahmenbedingungen eines Prozessfalles“* entstehen. Die Konsistenz kann durch die Funktion „Teilprozess anpassen“ wieder hergestellt werden, jedoch muss der Benutzer sie manuell ausführen (Schritt 15, entspricht Schritt 10). Die Schritte 11, 14 und 15 werden solange durchlaufen, bis das Projekt abgeschlossen ist. [Rupp02, 124f] Zur Ableitung eines kompletten Prozessmodells von den Projektrahmenbedingungen würde *„komplexere Konfigurationsalgorithmen und weiteres Konfigurationswissen“* benötigt werden. [Rupp02, 162]

Der Grad der Unterstützung für diese Phase ist eher niedrig einzustufen, da die Strategien der Wiederverwendung nicht über das Auswählen des Kontexts und das Suchen von Prozessbausteinen hinausgeht. Es werden keine Strategien zur Generierung von Lösungen zur Verfügung gestellt.

Bewertung: Niedrig

*Ansatz nach Krampe:*

Die Wiederverwendung besteht aus den folgenden fünf Aufgaben: *„Bestimmen, Auswählen und Kopieren von Teilentwürfen sowie der Konfiguration von Entwurfsschemata und dem Lesen von unbekannten Entwurfselementen“*. Die ersten vier Aufgaben erfolgen völlig automatisiert. Lediglich beim Lesen unbekannter Entwurfselemente erfolgt eine Interaktion mit dem Benutzer, wobei dieser in einen Dialog integriert wird, um zusätzliche Informationen bekanntzugeben (z.B. Entitätstyp: benötigte Attribute, Primärschlüssel, etc.). [Kram98, 100ff]

Der Benutzer wird fast vollständig von diesem Ansatz unterstützt und Strategien zur Generierung von Lösungen sind vorhanden.

Bewertung: Hoch

[22] Manuelle Anpassung bedeutet, dass die Editierung lediglich computer-gestützt erfolgt, „ohne inhaltliche Führung und Hilfestellung vom Anwendungssystem“ [Rupp02, 2]

***Kriterium: Grad der Unterstützung bei der kontrollierten Anpassung an den Anwendungskontext***

Der Grad der Unterstützung in dieser Phase wird durch Strategien bestimmt, die zur Anpassung an den Anwendungskontext verwendet werden. Es ist notwendig, dass die Anpassung kontrolliert vor sich geht.

*Ansatz nach Hagen:*

Die Anpassung an den aktuellen Anwendungskontext erfolgt manuell durch den Benutzer, und zwar durch das Instanziieren des Prozessmusters, also dem Hinzufügen von zusätzlichen Details. [Hage05, 205]

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Die Anpassung an den projektspezifischen Kontext (Anwendungskontext) erfolgt laut Rupprecht semi-automatisch. Unter semi-automatisch wird hier das systemseitige Generieren von Vorschlägen zur Individualisierung verstanden, die vom Benutzer angenommen bzw. abgelehnt werden können. [Rupp02, 3]

Damit ist das vorher erwähnte Auswählen der Projektrahmenbedingungen und Auswählen der Prozessbausteine gemeint.

Bewertung: Niedrig

*Ansatz nach Krampe:*

Dieser Ansatz übernimmt die Anpassung der Beziehungen vollständig [Kram98, 100ff] (vgl. Teilaufgabe *„Kopieren von Teilentwürfen und Anpassung der Lösung“*, Abbildung 137).

Bewertung: Hoch

***Kriterium: Interaktion mit Benutzer***

Dieses Kriterium ist wichtig, damit der Benutzer die Anpassung beeinflussen kann. Daher soll der Benutzer die vorgeschlagene Anpassung gegebenenfalls auch ablehnen können. Dieses Kriterium bestimmt daher, ob dies möglich ist.

*Ansatz nach Hagen:*

Der Benutzer führt die Anpassung manuell durch das Hinzufügen von zusätzlichen Details, also durch das Instanziieren der Prozessmuster durch [Hage05, 205].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Die Projektabwicklungsphase ist ebenfalls durch hohe Interaktion zwischen Benutzer und System gekennzeichnet [Rupp02, 123ff].

Bewertung: Hoch

*Ansatz nach Krampe:*

Der Benutzer kann die Anpassung an den aktuellen Kontext nicht beeinflussen, da er in dieser Phase lediglich zur Bekanntgabe neuer Informationen von unbekannten Elementen hinzugezogen wird [Kram98, 100ff].

Bewertung: Niedrig

***Kriterium: Flexibilität bei Änderung des Anwendungskontexts***

Ein weiteres wesentliches Kriterium ist die Flexibilität eines Werkzeugs. Änderungen der Anforderungen, und damit die Notwendigkeit Prozesse zu ändern, kommen in der Praxis häufig vor. Daher soll dieses Kriterium Auskunft darüber geben, ob bei einer Änderung des Anwendungskontexts der gesamte Prozess der Wiederverwendung von vorne begonnen werden muss oder ob das Vorgehensmodell des jeweiligen Ansatzes eine Flexibilität unterstützt und eine entsprechende Strategie anbietet.

*Ansatz nach Hagen:*

Ändert sich der Anwendungskontext, ändert sich auch das Problem, da dieses durch den initialen und den resultierenden Kontext bedungen ist. Daher muss zu dem geänderten Anwendungskontext ein neues Problem gesucht und daher ein neues Prozessmuster. [Hage05, 205f]

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Durch die hohe Interaktion dieses Ansatzes können Änderungen des Kontexts während der Projektlaufzeit durchgeführt werden (vgl. Abbildung 88 auf Seite 113).

Bewertung: Hoch

*Ansatz nach Krampe:*

Eine eventuelle Änderung des Anwendungskontexts bedeutet, dass mit dem CBR-Cycle entweder wieder von vorne begonnen werden müsste oder die Anpassungen manuell durchgeführt werden müssten. Anwendungskontext bedeutet bei diesem Ansatz das spezifische Unternehmensmodell, das durch Hinzufügen von weiteren Details zu einem Referenzmodell entstanden ist. Bei einer geringfügigen Änderung des Anwendungskontexts ist es möglich, dass eine Anpassung eines Unternehmensmodells weniger aufwendig ist, als *„ein Referenzmodell zu konkretisieren“*. [Kram98, 35f]

Bewertung: Niedrig

***Kriterium: Grad der Unterstützung beim Hinzufügen weiterer Details***

Abstrakte Artefakte können häufiger wiederverwendet werden, da sie schneller allgemein gültig sind als speziellere. Jedoch muss eine geeignete Strategie vorhanden sein, um das wiederzuverwendende Artefakt an den jeweiligen Anwendungskontext anzupassen und weitere Details hinzuzufügen. Zusätzlich soll durch eine Beziehung zwischen dem allgemeineren und spezielleren Artefakt eine Verbindung hergestellt werden, um Vererbung darstellen zu können, wobei diese konsistent gehalten werden sollen.

*Ansatz nach Hagen:*

Zusätzliche Details werden manuell hinzugefügt. Wird die neue Prozesslösung als Prozessmuster in den Prozessmusterkatalog aufgenommen, kann das abstrakte Prozessmuster über eine Refinement-Beziehung mit dem speziellen Prozessmuster verbunden werden. Es kann jedoch auch *„als völlig eigenständiges Prozessmuster“* aufgenommen werden. [Hage05, 205f]

Da die Details manuell hinzugefügt werden müssen, es jedoch die Möglichkeit gibt, diese beiden Prozessmuster über eine Beziehung zu verbinden, wird dieses Kriterium als teilweise erfüllt angesehen.

Bewertung: Mittel

*Ansatz nach Rupprecht:*

Es werden lediglich Vorschläge zum Einbau von bestehenden Prozessbausteinen bzw. Musterprozessen generiert. Dabei geschieht das Hinzufügen von weiteren Details, also das Konkretisieren von Prozessbausteinen, manuell. [Rupp02, 162] Durch die Generalisierungs- und Spezialisierungsbeziehung werden Prozessvarianten miteinander verbunden. Jedoch werden diese Prozessbausteine *„nicht konsistent gehalten“* und Änderungen eines Prozessbausteins führen zu keiner Änderung des verknüpften Prozessbausteins. Eine *„Vererbung von Strukturen und Eigenschaften“* im Sinne der Objektorientierung ist nicht vorgesehen. [Rupp02, 85]

Der Benutzer erhält keine Unterstützung beim Hinzufügen von weiteren Details.

Bewertung: Niedrig

*Ansatz nach Krampe:*

Innerhalb der Retrieve-Phase werden Entwurfsmerkmale angegeben, die benötigt werden. Dabei können auch unbekannte Elemente angegeben werden. Dazu musste in der Retrieve-Phase *„die Position des neuen Entwurfselements innerhalb der Indexbäume“* angegeben werden. Diese unbekannten Elemente werden in der Reuse-Phase eingelesen und der Benutzer wird bezüglich der benötigten Angaben befragt. Mit dieser Strategie können weitere Details hinzugefügt werden. Die Elemente werden somit in die Indexbäume an der entsprechenden Position eingefügt. [Kram98, 108f] Durch den Beziehungstyp innerhalb der Indexbäume (*is-a*) erfolgt eine Spezialisierung und auch Vererbung. [Kram98, 89]

Bewertung: Hoch

### 6.2.4 Überprüfung der angepassten Prozesslösung

***Kriterium: Konsistenzprüfung durch das System***
Dieses Kriterium soll über den Grad der Unterstützung durch den jeweiligen Ansatz bei der Konsistenzprüfung Auskunft geben. Hierbei soll geprüft werden, ob die jeweils definierten Regeln der Ansätze nach einer Anpassung der Prozesslösung automatisiert überprüft werden.

*Ansatz nach Hagen:*

Durch die Verwendung eines Prozessmusterkatalogs, der die Prozessmuster verknüpft durch Prozessmusterbeziehungen darstellt, werden *„Inkonsistenzen zwischen Prozessmustern“*

aufgedeckt, da formal definiert wird, wie die Prozessmuster zueinander in Beziehung stehen dürfen [Hage05, 54].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Dieser Ansatz basiert auf Gestaltungsregeln, mit dessen Hilfe die Prozessbausteine eingebaut werden. Von den projektspezifischen Rahmenbedingungen ausgehend werden weitere Rahmenbedingungen mit Hilfe der Kontextanpassungsregeln hinzugenommen. Ausgehend von den Kontextanpassungsregeln werden innerhalb der Problemlösungskomponente mit Hilfe des Regelinterpreters durch eine Vorwärtsverkettung Prozessanpassungsregeln gesucht und von diesen ausgehend pro Prozessanpassungsregel die entsprechenden Einbauregeln, die die Prozessbausteine dem Benutzer zum Einbau vorschlagen. Während des Ablaufes der Prozessindividualisierung können die Prozessmodellinstanzen manuell angepasst und weitere Projektrahmenbedingungen hinzugefügt werden. Diese Überprüfung der Konsistenz muss jedoch vom Benutzer angestoßen werden und geschieht nicht automatisch durch das System (mit der Funktion „Teilprozess anpassen“). [Rupp02, 116]

Da zwar eine Funktion zur Konsistenzprüfung vorhanden ist, diese aber durch den Benutzer ausgeführt werden muss, ist dieses Kriterium teilweise erfüllt.

Bewertung: Mittel

*Ansatz nach Krampe:*

In der Revise-Phase wird automatisiert durch diesen Ansatz der Entwurf auf Fehler überprüft und anhand der Konsistenzregeln auf Metamodellebene und Referenzmodellebene korrigiert [Kram98, 110f].

Bewertung: Hoch

***Kriterium: Überprüfung der angepassten Prozesslösung durch den Benutzer***

Eine Überprüfung durch das System reicht nicht aus. Daher muss auch eine Überprüfung der angepassten Prozesslösung durch den Benutzer ebenfalls möglich sein. Dieses Kriterium soll darüber Auskunft geben.

*Ansatz nach Hagen:*

Da der Benutzer die Instanziierung des Prozessmusters völlig übernimmt [Hage05, 205], impliziert dies auch eine zusätzliche Überprüfungsmöglichkeit durch den Benutzer.

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Wegen der hohen Interaktion mit dem Benutzer und der weitgehend manuellen Anpassung des Anwendungskontexts (Rahmenbedingungen) bzw. der Prozessmodellinstanzen (vgl. weiter oben) ist dieses Kriterium erfüllt.

Bewertung: Hoch

*Ansatz nach Krampe:*

Die angepassten und von diesem Ansatz automatisiert überprüften Entwürfe können innerhalb der Revise-Phase vom Benutzer verändert werden (externe Prüfung). Insbesondere betrifft dies Änderungen bzw. Definitionen von Modellen und Komponenten. [Kram98, 126ff]

Bewertung: Hoch

### 6.2.5 Integration neuer Prozesslösungen in die Wissensbasis

***Kriterium: Dokumentation von erprobtem Wissen***

Dieses Kriterium liefert eine Aussage darüber, ob das Artefakt sich in dem jeweiligen Anwendungskontext „erprobt" hat. Dabei ist es notwendig, dass der Status der Qualität bzw. der Erfahrungswert des Artefakts angegeben werden kann.

*Ansatz nach Hagen:*

Nach der Prozessmusteranwendung kann der Benutzer das Prozessmuster nach unterschiedlichen Kriterien bewerten. Dabei erwähnt Hagen Kriterien wie Verständlichkeit, eine *„korrekte Darstellung des Kontexts und des Prozesses"* und fehlende Beziehungen zwischen Prozessmustern. Der Status bzw. der Erfahrungswert kann bei diesem Ansatz nicht explizit angegeben werden, aber die Erfahrungen werden nach der Prozessmusteranwendung an den Pattern Designer weitergeleitet, der die Prozessmuster entsprechend ändert. [Hage05, 206]

Der Qualitätsstatus bzw. Erfahrungswert der Prozessmuster kann nicht angegeben werden, sondern wird an den Pattern Designer kommuniziert.

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Der Erfahrungswert des Prozessbausteins (dokumentierten Wissens) kann durch die Ausprägungen „Best Practice" und „Lessons Learned" angegeben werden. Der Qualitätsstatus kann durch die Begriffe „Entwurf", „freigegeben", „abgestimmt" bzw. „Qualitätsbaustein" vergeben werden. [Rupp02, 83]

Dieses Kriterium ist erfüllt, jedoch fehlt eine eindeutige Definition des Unterschieds zwischen „freigegeben", „abgestimmt" und „Qualitätsbaustein" bei dem Qualitätsstatus.

Bewertung: Hoch

*Ansatz nach Krampe:*

Während der Retain-Phase kann der Benutzer die *„Anwendbarkeit des Entwurfs"* beurteilen. Dabei wird das in Tabelle 7 dargestellte Schema verwendet (Seite 142). Unter dem Attribut *status* kann der Benutzer den Status des Entwurfs angeben („learned", „work in progress", „has-to-be-justified") [Kram98, 139] Zusätzlich muss der Anwendungskontext für das Fallbeispiel festgehalten, in dem dieses verwendet werden kann. Dieser Ansatz dient dazu, um häufig wiederkehrende Aufgaben (Routine-Aufgaben), z.B. *„Suche, Vergleich, Auswahl und Anpassung von Entwurfselementen"* zu übernehmen und die Benutzer *„bei der Strukturierung von Entwurfserfahrung"* zu unterstützen. [Kram98, 163ff]

Bewertung: Hoch

***Kriterium: Grad der Unterstützung bei der Einordnung des wieder zu verwendenden Artefakts in die Wissensbasis***

Hiermit wird der Grad der Unterstützung bei der Einordnung in die Wissensbasis bestimmt, indem überprüft wird, ob Strategien zur Anpassung der Beziehungen zwischen den Artefakten zur Verfügung gestellt werden. Je schlechter die Strategie zur Einordnung in die Wissensbasis ist, desto größer ist die Gefahr eines Wildwuchses der Wissensbasis, was erstens die Suche verkomplizieren und zweitens die Gefahr erhöhen könnte, dass gleiche Artefakte des Öfteren gespeichert würden (Gefahr der Inkonsistenz).

*Ansatz nach Hagen:*

Hagen weist diese Aufgabe dem Pattern Designer zu. Dieser dokumentiert neue Prozessmuster, sofern sich diese für die Wiederverwendung als geeignet erweisen (*„Rule of Three“* nach [Copl94]). Unterstützung erhält der Pattern Designer, indem er zur Dokumentation die Prozessmusterschemata und Problemschemata verwendet [Hage05, 65ff]. Zur Einordnung in den Prozessmusterkatalog kann er auf die bestehenden Problem- und Prozessmusterdiagrame zurückgreifen und die neuen Prozessmuster in den Prozessmusterkatalog einordnen (vgl. Abschnitt 3.4.1 und 3.4.5).

Es ist eine klare Vorgehensweise definiert wird, jedoch muss der Benutzer die Beziehungen selbst definieren.

Bewertung: Mittel

*Ansatz nach Rupprecht:*

Die Wissensbasis besteht bei diesem Ansatz aus einer Ordnerstruktur für generische Prozessmodelle und aus einer Ordnerstruktur für generische Rahmenbedingungen. Die obersten Ebenen sind deduktiv (*„d.h. abgeleitet aus validen Theorien und Hypothesen“*) und die unteren Ebenen können von den Benutzern selbst gestaltet werden. [Rupp02, 84f] Diese generischen Prozessmodelle können auch in mehreren Ordnern (Prozesskategorien) abgelegt werden [Rupp02, 95]. Auch die Rahmenbedingungen können mehreren Einflussgrößenkategorien zugeordnet werden [Rupp02, 77]. Rupprecht schlägt vor, dass die generischen Rahmenbedingungen, die generischen Prozessmodelle und die Prozesskategorien, die Prozessmodelle enthalten können, anhand von Namenskonventionen benannt werden sollen, so dass eine Speicherung von gleichen oder ähnlichen Objekten verhindert wird. [Rupp02, 87]

Dieser Ansatz unterstützt den Benutzer gering bei der Einordnung des wieder zu verwendenden Artefakts in die Wissensbasis und die Gefahr des Wildwuchses ist groß.

Bewertung: Niedrig

*Ansatz nach Krampe:*

Nachdem das Fallbeispiel erstellt und einer externen Prüfung unterzogen wurde (sich in der *„realen Welt bewährt“* hat oder nicht), wird es in die Entwurfsbibliothek eingeordnet. Dabei müssen eventuell Konsistenzregeln sowie Entwurfsschemata und anschließend die Indexkategorien angepasst werden. Dieser Ansatz unterstützt bei der Anpassung der Konsistenzregeln bzw. Entwurfsschemata lediglich das Modifizieren des *„Methoden- und Domänenwissens“* durch Änderung des Prolog-Quellcodes. Die Indexkategorien können einfach durch den Web-Browser realisiert werden. [Kram98, 163ff]

Die Einordnung in die Entwurfsbibliothek wird teilweise durch diesen Ansatz unterstützt.

Bewertung: Mittel

| | **Ansätze** | | |
|---|---|---|---|
| **Kriterien** | **Hagen** | **Rupprecht** | **Krampe** |
| **Allgemein** | | | |
| *Begriffsdefinition* | hoch | niedrig | hoch |
| *Methodenunabhängigkeit* | niedrig | mittel | hoch |
| *Domänenunabhängigkeit* | hoch | hoch | mittel |
| ***Integration von externen Prozesslösungen*** | | | |
| *Strukturierter Aufbau der Wissensbasis* | hoch | mittel | hoch |
| *Unterschiedliche Abstraktionsstufen* | hoch | hoch | hoch |
| *Berücksichtigung von Referenzlösungen* | niedrig | niedrig | hoch |
| *Atomare Zerlegung eines Prozessmodells* | hoch | hoch | hoch |
| *Einheitliche Beschreibung der wieder zu verwendenden Artefakte* | hoch | niedrig | hoch |
| *Beschreibung der Beziehungen zwischen den wieder zu verwendenden Artefakten* | hoch | hoch | hoch |
| *Formale Beschreibung der Prozesslösungen der wieder zu verwendenden Artefakte* | hoch | hoch | hoch |
| *Grad der automatisierten Unterstützung beim Befüllen der Wissensbasis* | mittel | mittel | niedrig |
| ***Suche nach geeigneten Prozesslösungen*** | | | |
| *Grad der Unterstützung bei der Suche nach Artefakten* | hoch | hoch | hoch |
| *Grad der Unterstützung bei der Auswahl der Artefakte* | niedrig | niedrig | hoch |
| *Interaktion mit Benutzer* | hoch | hoch | niedrig |
| ***Anwendung von gefundenen Prozesslösungen*** | | | |
| *Grad der Unterstützung bei der Wiederverwendung der gefundenen Prozesslösungen* | niedrig | niedrig | hoch |
| *Grad der Unterstützung bei der kontrollierten Anpassung an den Anwendungskontext* | niedrig | niedrig | hoch |
| *Interaktion mit Benutzer* | hoch | hoch | niedrig |
| *Flexibilität bei Änderung des Anwendungskontexts* | niedrig | hoch | niedrig |
| *Grad der Unterstützung beim Hinzufügen weiterer Details* | mittel | niedrig | hoch |
| ***Überprüfung der angepassten Prozesslösungen*** | | | |
| *Konsistenzprüfung durch das System* | hoch | mittel | hoch |
| *Überprüfung der angepassten Prozesslösung durch den Benutzer* | hoch | hoch | hoch |

| | | | |
|---|---|---|---|
| ***Integration neuer Prozesslösungen in die Wissensbasis*** | | | |
| *Dokumentation von erprobtem Wissen* | niedrig | hoch | hoch |
| *Grad der Unterstützung bei der Einordnung des wieder zu verwendenden Artefakts in die Wissensbasis* | mittel | niedrig | mittel |

**Tabelle 14: Bewertungsmatrix konzeptionelle Gegenüberstellung**

## *6.3 Prototyp*

In diesem Abschnitt wird die Umsetzung der jeweiligen Ansätze mit den realisierten Prototypen verglichen. Dabei werden Kriterien angeführt, die sich auf diese Realisierung beziehen.

***Kriterium: Schnittstellen zu anderen Modellierungswerkzeugen***

Um die volle Effizienz der reaktiven Wiederverwendung ausnutzen zu können, müssen Schnittstellen zu anderen Modellierungswerkzeugen bestehen. Sie dienen dazu, bereits erstellte Prozessmodelle in die Wissensbasis zu importieren und für die weitere Wiederverwendung dem jeweiligen Ansatz entsprechend aufzubereiten.

*Ansatz nach Hagen:*

Es ist kein Import von bereits vorhandenen Prozessmodellen möglich [Hage05]. Die Dokumentation von Prozessmustern erfolgt wie in Abschnitt 3.4.1 dargestellt.

Bewertung: Niedrig

*Ansatz nach Rupprecht:*

Der Wissenserwerb des generischen Wissens (generische Rahmenbedingungen, Gestaltungsregeln und Prozessmodelle) muss durch den Experten mit Hilfe der Wissenserwerbskomponente manuell durchgeführt werden. Es werden hierzu Funktionen zur Verfügung gestellt (vgl. Abschnitt 4.3.1). Es sind keine Strategien vorgesehen, die die automatische Aufbereitung des generischen Wissens (generische Prozessmodelle, Rahmenbedingungen und Gestaltungsregeln) vornehmen. Rupprecht erwähnt selbst, dass Forschungsbedarf in der *„Ableitung von generischen Strukturen und Abhängigkeiten aus Prozessfällen“* und der *„Transformation von implizitem Wissen in großen Datenbeständen zu explizitem Wissen“* besteht. [Rupp02, 163]

Dieser Ansatz unterstützt keinen Import von Prozessmodellen von anderen Modellierungswerkzeugen und keine anschließende automatische Aufbereitung für die Bibliothek.

Bewertung: Niedrig

*Ansatz nach Krampe:*

Über das ARIS-Toolset gibt es auch die Möglichkeit spezifische Unternehmensmodelle in das Repository des CBModelers zu importieren. Der Datenaustausch erfolgt dabei über APIs zu anderen CASE- bzw. Workflow-Tools. Diese Unternehmensmodelle können anschließend als Fallbeispiele für die Reaktive Wiederverwendung genutzt werden. [Kram98, 132ff]

Bewertung: Hoch

***Kriterium: Visualisierung der vorgeschlagenen Artefakte zur Entscheidungsunterstützung***

Als Entscheidungsunterstützung kann auch eine Visualisierung dienen. Unter Visualisierung der vorgeschlagenen Artefakte ist hier gemeint, dass das auszuwählen mögliche Artefakt entsprechend der eingesetzten Modellierungsmethode in der entsprechenden Notation grafisch dargestellt wird.

*Ansatz nach Hagen:*

Die Visualisierung der Prozesslösungen wurde bei der Implementierung des Prototyps durch JGraph 2.0 realisiert [Hage05, 237] und wird in der Anforderungsspezifikation bei den nicht-funktionalen Anforderungen angeführt [Hage05, 180].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Mit Hilfe eines Prozess-Editors kann die Lösung eines ausgewählten Prozessbausteins bzw. Musterprozesses visualisiert werden [Rupp02, 143].

Bewertung: Hoch

*Ansatz nach Krampe:*

Dieses Kriterium ist in der Retrieve-Phase nicht wichtig, da der Ansatz eine Priorisierung der Fallbeispiele mit Hilfe der Ähnlichkeitsbestimmung vornimmt, ohne dass der Benutzer eingreifen kann [Kram98, 89ff].

Bewertung: Nicht notwendig

***Kriterium: Visualisierung der angepassten Artefakte***

Damit die Modifikation auf ihre Korrektheit überprüft bzw. erleichtert werden kann, ist eine Visualisierung der angepassten Artefakte hilfreich. Unter Visualisierung wird hier eine grafische Darstellung der endgültigen Lösung verstanden.

*Ansatz nach Hagen:*

Durch die Umsetzung der nicht-funktionalen Anforderung einer grafischen Benutzerschnittstelle mit JGraph 2.0 [Hage05, 180] können auch Prozesslösungen der angepassten Artefakte betrachtet werden.

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Der ausgewählte Prozessbaustein bzw. Musterprozess kann im Baustein-Editor angezeigt und auch bearbeitet werden [Rupp02, 143].

Bewertung: Hoch

*Ansatz nach Krampe:*

Krampe schlägt bereits anstatt seines ausschließlich auf HTML basierten Clients *„eine leistungsfähigere Benutzerschnittstelle“* vor, wie etwa mit Java oder ActiveX controls. Diese sollte anschließend die Entwürfe nicht nur textuell, sondern auch grafisch darstellen können. [Kram98, 180]

Bewertung: Niedrig

***Kriterium: Verwendung als Dokumentationswerkzeug***

Dieses Kriterium dient der Diskussionsgrundlage zwischen Auftragnehmer bzw. Benutzer und dem Auftraggeber bzw. den Anwendern. Es soll die Auswahl der Artefakte wesentlich erleichtern und Missverständnisse bereits im Vorfeld ausräumen.

*Ansatz nach Hagen:*

Der Prototyp, der diesen Ansatz umsetzt, kann die Prozessmuster und ihre Prozessmusterbeziehungen als Kommunikationsgrundlage darstellen [Hage05, 68].

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Eine Verwendung als Dokumentationswerkzeug ist bei diesem Ansatz nicht vorgesehen [Rupp02].

Bewertung: Niedrig

*Ansatz nach Krampe:*

Da dieser Prototyp einen Web-Browser als Client verwendet und die Entwicklungsumgebung durch Web-Technologie ergänzt wird, und zwar durch HTML-basierten Werkzeugansatz, kann dieses Werkzeug ebenfalls gut für die Dokumentation verwendet werden. [Kram98, 128f]

Bewertung: Hoch

***Kriterium: Ort- und Plattformunabhängigkeit***

Der Prozess der reaktiven Wiederverwendung soll auch Teamfähigkeit unterstützen. Daher ist Ort- und Plattformunabhängigkeit ein wichtiges Kriterium, das in allen Phasen der reaktiven Wiederverwendung gegeben sein soll.

*Ansatz nach Hagen:*

Als Plattform dient die Process Pattern Workbench, die als Prototyp und mit JDK 1.3.1 und J2EE 1.3.1 realisiert wurde. Damit sollen alle Aufgaben des Process Pattern Managements erledigt werden. Eine der nicht-funktionalen Anforderungen ist *„Teamfähigkeit der Software“*. Diese Anforderung soll *„Versionskonflikte und Inkonsistenzen“* der definierten

Prozessmuster vermeiden. Daher werden diese *„in einem zentralen Katalog abgelegt“*. [Hage05, 180ff]

Bewertung: Hoch

*Ansatz nach Rupprecht:*

Mit POWM (Prozessorientiertes Wissensmanagement) als Werkzeug wird eine verteilte Prozessmodellierung, Prozessmodellierung durch Endbenutzer (ohne Expertenwissen für Prozessmodellierung) sowie Dekomposition von Prozessmodellen unterstützt. Dieses Werkzeug ist eine plattformunabhängige Java-Anwendung und verfügt über eine Client-Server-Architektur. [Rupp02, 126ff]

Bewertung: Hoch

*Ansatz nach Krampe:*

Da bei diesem Ansatz die Entwicklungsumgebung durch Web-Technologie ergänzt wird, und zwar durch *„einen rein HTML-basierten Werkzeugansatz“*, ist dieses Werkzeug ort- und plattformunabhängig [Kram98, 128f].

Bewertung: Hoch

| | **Ansätze** | | |
|---|---|---|---|
| **Kriterien** | **Hagen** | **Rupprecht** | **Krampe** |
| *Schnittstellen zu anderen Modellierungswerkzeugen* | niedrig | niedrig | hoch |
| *Visualisierung der vorgeschlagenen Artefakte zur Entscheidungsunterstützung* | hoch | hoch | nicht notwendig |
| *Visualisierung der angepassten Artefakte* | hoch | hoch | niedrig |
| *Verwendung als Dokumentationswerkzeug* | hoch | niedrig | hoch |
| *Ort- und Plattformunabhängigkeit* | hoch | hoch | hoch |

**Tabelle 15: Bewertungsmatrix Prototyp**

## *6.4 Zusammenfassung*

Krampe unterstützt den Prozess der reaktiven Wiederverwendung durch seinen Ansatz durch die zur Verfügung gestellten Strategien optimal und liefert auch generische Lösungen als Ergebnis, indem die Beziehungen zwischen den eingefügten Prozessmodellen automatisiert angepasst werden. Jedoch ist dieser Ansatz durch den CBR-Cycle weniger flexibel, da bei Änderungen des Anwendungskontexts dieser von vorne begonnen werden muss. Die prototypische Umsetzung des Konzepts weist Schnittstellen zu ARIS-Toolset auf, um existierende Prozessmodelle in das Repository zu importieren. Eine Visualisierung der vorgeschlagenen und angepassten Artefakte findet bei der momentanen Umsetzung nicht statt, sondern wird bei den angepassten Artefakten textuell angezeigt.

Der Ansatz nach Hagen unterstützt optimal die Strukturierung der Wissensbasis durch die formalen Definitionen, indem den verwendeten Konzepten syntaktische und semantische Bedeutung zugewiesen wird. Dadurch werden die Suche und die Auswahl der Artefakte wesentlich unterstützt. Lediglich bei der Auswahl von Artefakten findet keine Priorisierung von vorgeschlagenen Artefakten statt, sondern müssen durch selbst gewählte Kriterien durch den Benutzer ausgewählt werden. Es fehlen Strategien, um Lösungen aus den gefundenen Prozessmustern generisch zusammenzustellen. Jedoch werden bei der prototypischen Umsetzung die vorgeschlagenen und angepassten Artefakte visualisiert, wodurch der Benutzer die Überprüfung der ausgewählten und erstellten Prozessmodelle selbst vornehmen kann. Eine Schnittstelle zu anderen Modellierungswerkzeugen wird nicht explizit angeführt. Eine durchgängige Unterstützung durch diesen Ansatz erfolgt daher nicht.

Der Ansatz nach Rupprecht stellt Funktionen zur Verfügung, mit dessen Hilfe unter anderem Prozessbausteine und Regeln definiert werden können und anschließend in einer Bibliothek von Rahmenbedingungen und generischen Prozessmodellen nach Prozesslösungen gesucht werden kann. Die Bibliotheken werden nicht eindeutig strukturiert, da Prozessbausteinen in mehreren Prozesskategorien, sowie Prozesskategorien auch in mehreren Prozesskategorien enthalten sein können. Durch die Regeln werden die Suche und die Auswahl nach Kriterien, die der Benutzer selbst wählt, wesentlich unterstützt. Auch bei diesem Ansatz werden keine Konzepte zur Priorisierung der vorgeschlagenen Artefakte zur Verfügung gestellt. Eine durchgängige Unterstützung durch diesen Ansatz erfolgt somit nicht.

# 7 Schlussbetrachtung

Zielsetzung der vorliegenden Arbeit war, ausgewählte Ansätze der reaktiven Wiederverwendung zu analysieren und zu vergleichen und dabei die Stärken und Schwächen der Konzepte und realisierenden Prototypen zu identifizieren. Dazu wurde aus den unterschiedlichen Klassen jeweils ein Vertreter ausgewählt.

In Kapitel 2 wurden die benötigten Grundlagen erklärt. Daher wurde die Prozessmodellierung erklärt und auf verschiedene Einsatzbereiche bezug genommen, um die Notwendigkeit der Prozessmodellierung aufgrund der verschiedenartigen Einsatzbereiche zu untermauern.

Die ausgewählten Ansätze, und zwar der musterbasierte Ansatz nach Hagen, der bausteinbasierte Ansatz nach Rupprecht und der fallbasierte Ansatz nach Krampe, wurden in den Kapitel 3, 4 und 5 eingehend untersucht. Dazu wurde ein klein gehaltenes Rahmenbeispiel auf die Ansätze angewendet, um die durch den gesamten Prozess der reaktiven Wiederverwendung zur Verfügung gestellte Unterstützung zu veranschaulichen.

In Kapitel 6 wurden Anforderungen an die reaktive Wiederverwendung aus der Literatur abgeleitet und in Kriterien überführt, die dem Konzept und dem Prototypen zugeordnet wurden. Durch diese Vorgehensweise konnten die Stärken und Schwächen der Konzepte und Prototypen der jeweiligen Ansätze identifiziert werden, die in Abschnitt 6.4 zusammengefasst wurden.

# Literaturverzeichnis


[AaPl94] Aamodt A., Plaza E.: *Case-Based Reasoning: Foundational issues, methodological variations and system approach.* AI Communications, 7(1): 39-59, 1994.

[AlBa96] Althoff K.-D., Bartsch-Spörl B.: *Decision Support for Case-Based Applications.* Wirtschaftsinformatik, 38(1):8-16, 1996.

[Ambl98] Ambler S.W.: *An Introduction to Process Patterns*. Link: http://www.ambysoft.com/downloads/processPatterns.pdf (Zugriff am 22.02.07).

[BaBS94] Bakhtari S., Bartsch-Spörl B.: *Bridging the Gap Between AI Technology and Design Requirements.* In: GERO, John S. und Sudweeks F (Herausgeber): *Proceedings of Artificial Intelligence in Design 1994*, Seiten 753-768. Kluwer Academic Publishers, 1994.

[Baum90] Baumgarten B.: *Petri-Netze, Grundlagen und Anwendungen.* B.I. Wissenschaftsverlag, 1990.

[BBCD04] Becker J., Brelage C., Crisandt J., Dreiling A., Holten R., Ribbert M., Seidel S.: *Methodische und technische Integration von Daten- und Prozessmodellierungstechniken für Zwecke der Informationsbedarfsanalyse.* Universität Münster, Arbeitsbericht Nr. 103, Münster, März 2004.

[BDKK02] Becker J. et al.: *Konfigurative Referenzmodellierung*. In: Becker J., Knackstedt R. (Hrsg.): Wissensmanagement mit Referenzmodellen – Konepte für die Anwendungssystem- und Organisationsgestaltung. Berlin et al. 2002, S.25-140.

[Beck96] Becker J.: *Handelsinformationssysteme.* Landsberg am Lech 1996.

[Beck98] Becker J.: *Die Architektur von Handelsinformationssystemen.* In: Ahlert D., Becker J., Olbrich R., Schütte R. (Hrsg.): Informationssysteme für das Handelsmanagement. Konzepte und Nutzung in der Unternehmenspraxis. Berlin u.a. 1998, S. 65-108.

[BeKn02] Becker J., Knackstedt R.: *Referenzmodellierung 2002 Methoden – Modelle – Erfahrungen.* Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 90, Westfälische Wilhelm-Universität Münster, 2002.

[BeKR05] Becker J., Kugeler M., Rosemann M. (Hrsg.): *Prozessmanagement. Ein Leitfaden zur prozessorientierten Organisationsgestaltung.* 5., überarbeitete und erweiterte Auflage. Springer. Berlin et al. 2005.

[BeRo96] Becker J., Rosemann M.: *Workflowmanagement – State-of-the-Art aus Sicht von Theorie und Praxis.* Link: http://www.wi.uni-muenster.de/inst/arbber/ab47.pdf (Zugriff am 22.02.2008).

[BeSc96] Becker J., Schütte R.: *Handelsinformationssysteme.* verlag moderne industrie, Landsberg/Lech, 1996.

[BHKS00] Becker J., Holten R., Knackstedt R., Schütte R.: *Referenz-Informationsmodellierung.* In: Bodendorf F., Grauer M. (Hrsg.): Verbundtagung Wirtschaftsinformatik 2000. Aachen 2000, S 86-109. http://www.wi.uni-muenster.de/improot/is/pub_imperia/doc/854.pdf (Zugriff am 28.02.2008).

[Blei91] Bleicher K.: *Organisation*. Formen und Modelle. Gabler, Wiesbaden 1991.

[BoDö01] Bortz J., Döring N.: *Forschungsmethoden und Evaluation für Human- und Sozialwissenschftler*, 3. Auflage Springer-Verlag Berlin Heidelberg New York, ISBN 3-540-41940-3, 2001.

[BöKS06] Bögl A., Kobler M., Schrefl M.: *Wiederverwendung von Prozessmodellen*. Wirtschaftsinformatik als Schlüssel zum Unternehmenserfolg. Festschrift anlässlich des 60. Geburtstages von o. Univ.-Prof. Dr. Friedrich Roithmayr. In: Kerstin Fink, Christian Ploder (Hrsg.), Deutscher Universitäts-Verlag (DUV), S. 137-152, 2006, ISBN: 3-8350-0293-7.

[Broc03] vom Brocke J.: *Referenzmodellierung. Gestaltung und Verteilung von Konstruktionsprozessen.* Advances in Information Systems and Management Science, Bd. 4, Logos-Verlag, Berlin, 2003.

[Chen76] Chen P.: *The Entity-Relationship Model – Toward a Unified View of Data*. ACM Transaction on Database Systems, Vol. 1, No. 1, März 1976, S.9-36.

[Copl94] Coplien J.: *A Development Process Generative Pattern Language.* In: Proceedings of PLoP, 1994.

[Copl95] Coplien J.: *A generative development-process pattern language.* ACM Press/Addison-Wesley Publishing Co., New York, USA. 1995.

[Copl96] Coplien J.: *Software Patterns.* SIGS Book & Multimedia, 1996.

[FeLo04] Fettke P., Loos P.: *Referenzmodellierungsforschung.* In: WIRTSCHAFTSIN-FORMATIK 46 (2004) 5, S.331-340.

[Floy92] Floyd C.: *STEPS-Projekthandbuch.* Universität Hamburg, 1992.

[FrHH00] Frøkjær H., Hertzum M., Hornbæk K.: *Measuring usability: are effectiveness, efficiency, and satisfaction really correlated?* In: Proceedings of the SIGCHI conference on Human factors in computing systems, 2000.

[Gada05] Gadatsch A.: *Grundkurs Geschäftsprozessmanagement*. 4. Auflage, Mercedes Druck, Berlin, 2005.

[GeHK99] Gerber S., Hiestermann A., Kittlaus H.-B.: *Management von Prozeßmodellen dezentraler BPR-Projekte mit Hilfe eines zentralen Referenzprozeßmodells.* 4.

Internationale Tagung Wirtschaftsinformatik, Universität des Saarlandes, Saarbrücken, 1999.

[GMPR01a] Gnatz M., Marschall F., Popp G., Rausch A., Schwerin W.: *Towards a Living Software Development Process based on Process Patterns*, Technische Universität München, 2001.

[GMPR01b] Gnatz M., Marschall F., Popp G., Rausch A., Schwerin W.: *Modular Process Patterns supporting an Evolutionary Software Development Process.* In: Proceedings of the 3rd Int. Conference on Product Focused Software Development, 2001.

[GrMü06] Gronau N., Müller C.: *Grundlagen der Geschäftsprozessmodellierung (EPK). Geschäftsprozessmanagement.* http://wi.uni-potsdam.de/homepage/potsdam.nsf/0/95E868B3117B6C80C12571540052351B/$File/VL-GPM_EPK.pdf (Zugriff am 31.07.2007).

[HaCh94] Hammer M., Champy J.: *Business Reengineering*. 2. Auflage, Frankfurt, New York 1994.

[Hage05] Hagen M.: *Definition einer Sprache zur Beschreibung von Prozessmustern zur Unterstützung agiler Softwareentwicklungsprozesse*, Dissertation, Universität Leipzig, 2005.

[Hamm90] Hammer M.: *Reengineering Work: Don't Automate, Obliterate*. In: Harvard Busines Review, Vol. 68, Nr. 4: 104-112, 1990.

[Hars94] Hars A.: *Referenzdatenmodelle – Grundlagen effizienter Datenmodellierung*. Wiesbaden 1994.

[HeHR04] Heinrich L.J., Heinzl A., Roithmayr F.: *Wirtschaftsinformatik-Lexikon*. 7. Auflage, München/Wien, 2004.

[Hess96] Hess T.: *Entwurf betrieblicher Prozesse: Grundlagen, bestehende Methoden, neue Ansätze*. Dissertation, Wiesbaden 1996.

[Holt00] Holten R.: *Entwicklung einer Modellierungstechnik für Data Warehouse-Fachkonzepte.* In: Schmidt H. (Hrsg.): Modellierung betrieblicher Informationssysteme. Proceedings der MobIS-Fachtagung 2000, 11. und 12. Oktober 2000, Siegen. Rundbrief der GI-Fachgruppe 5.10, 7 (2000) 1, S.3-21.

[KeNS92] Keller G., Nüttgens M., Scheer A.-W.: *Semantische Prozessmodellierung auf der Grundlage „Ereignisgesteuerter Prozessketten (EPK)"*. In: Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes, 1992.

[Kram96] Krampe D.: *Ein wissensbasiertes Werkzeug zur Unterstützung des konzeptionellen Entwurfs betrieblicher Informationssysteme*. WWZ-Discussion Paper 9604, Universität Basel, 1996.

[Kram98] Krampe D.: *Fallbasierter konzeptioneller Informationssystementwurf.* Dissertation, Universität Basel, 1998.

[KrLu96] Krampe D., Lusti M.: *Towards a Case-Based Assistant fort he Conceptual Modelling of Information Systems.* In: *Information and Classification: Proceedings of the 20th Annual Conference of the "Gesellschaft für Klassifikation e.V."*, Universität Freiburg im Breisgau. Spinger Verlag, März 1996.

[Küpp82] Küpper H.-U.: *Ablauforganisation*. Fischer, Stuttgart/New York, 1982.

[Lang97] Lang K.: *Gestaltung von Geschäftsprozessen mit Referenzprozessbausteinen*. Dissertation. Gabler, Wiesbaden 1997.

[LiKo06] List B., Korherr B.: *An Evaluation of Conceptual Business Process Modelling Languages.* SAC '06, Dijon, Frankreich, April 23-27, 2006.

[Meff91] Meffert H.: *Marketing. Grundlagen der Absatzpolitik.* 7. Auflage, Wiesbaden 1991.

[Meis01] Meise V.: *Ordnungsrahmen zur prozessorientierten Organisationsgestaltung. Modelle für das Management komplexer Reorganisationsprojekte.* Hamburg 2001.

[Nobl98] Noble J.: *Towards a Pattern Language for Object Oriented Design.* In: Proceedings of the Technology of Object-Oriented Languages and Systems, IEEE Computer Society, Washington DC, 1998.

[NüRu02] Nüttgens M., Rump J.F.: *Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK).* In: Desel J., Weske M. (Hrsg.): *Promise 2002 – Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen*. Proceedings des GI-Workshops und Fachgruppentreffens (Potsdam, Oktober 2002), LNI Vol. P-21, Bonn 2002, S. 64-77.

[Öste95] Österle H.: *Business Engineering*. Prozess- und Systementwicklung. Entwurfstechniken. Band 1, Berlin 1995.

[Remm97] Remme M.: *Konstruktion von Geschäftsprozessen: ein modellgestützter Ansatz durch Montage generischer Prozeßpartikel*. Dissertation, Universität des Saarlandes. Gabler, Wiesbaden 1997.

[Rich03] Richter M.: *Fallbasiertes Schließen*. In: Görz G. Rollinger C.-R., Schneeberger J. (Hrsg.): *Handbuch der Künstlichen Intelligenz*. 4. Auflage, München/Wien, 2003. S. 407-430.

[Rohl07] Rohloff M.: *Ein Referenzmodell für die Prozesse der IT-Organisation.* HMD Praxis der Wirtschaftsinformatik, Heft 256, 2007.

[Rose02] Rosenkranz F.: *Geschäftsprozesse.* Modell- und computergestützte Planung. Springer-Verlag, Berlin Heidelberg 2002.

[RuPR99] Rupprecht C., Peter G., Rose T.: *Ein modellgestützter Ansatz zur kontextspezifischen Individualisierung von Prozessmodellen.* Link: http://www-ai.cs.uni-magdeburg.de/~gerhard/Publications/wi99.pdf (Zugriff am 06.11.2007).

[RRFS01] Rupprecht C., Rose T., Fünffinger M., Schott H., Sieper A., Schlick C., Mühlfelder M.: *Management von Prozesswissen in Fahrzeugentwicklungs-projekten*. In: Schnurr H.-P. et al. (Hrsg.): *Professionelles Wissensmanagement – Erfahrungen und Visionen*, Tagungsband der 1. Konferenz Professionelles Wissensmanagement, 14.-16. März 2001, Shaker, Baden-Baden 2001, S. 29-33.

[Rupp02] Rupprecht C.: *Ein Konzept zur projektspezifischen Individualisierung von Prozessmodellen*, Dissertation, Universität Karlsruhe, 2002.

[Sche96] Scheer A.-W. (Hrsg.): *Business Process Reengineering in der Verwaltung.* Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes. Link: http://www.iwi.uni-sb.de/Download/iwihefte/heft129.pdf (Zugriff am 29.02.2008).

[Sche97] Scheer A.-W.: *Wirtschaftsinformatik. Referenzmodelle für industrielle Geschäftsprozesse.* Springer Berlin et al. 1997.

[Sche01] Scheer A.-W.: *ARIS – Modellierungsmethoden, Metamodelle, Anwendungen.* Institut für Wirtschaftsinformatik, Universität des Saarlands, 2001.

[Schu01] Schulze D.: *Grundlagen der wissensbasierten Konstruktion von Modellen betrieblicher Systeme*, Dissertation, Universität Bamberg, 2001. ISBN 3-8265-8939-4.

[Schü98] Schütte R.: *Grundsätze ordnungsmäßiger Referenzmodellierung – Konstruktion konfigurations- und anpassungsorientierter Modelle*. Gabler, Wiesbaden 1998.

[ScBe98] Schütte R., Becker J.: *Subjektivitätsmanagement bei Informationsmodellen*. In: Proceedings of the Modellierung ´98: Arbeitsberichte Angewandte Mathematik und Informatik. Bericht Nr. 6/98-I. Münster. 1998. S. 81-86.

[Schw99] Schwegmann A.: *Objektorientierte Referenzmodellierung. Theoretische Grundlagen und praktische Anwendung.* Wiesbaden 1999.

[ScSt98] Schulte H., Stumme G.: *Projektmanagement.* In: Kleinkamp M., Plinke W. (Hrsg.): *Auftrags- und Projektmanagement – Projektbearbeitung für den technischen Vertrieb*, Springer, Berlin et al. 1998, S. 227-266.

[Seid02] Seidl J.: *Business Process Performance. Modellbezogene Beurteilung und Ansätze zur Optimierung.* HMD Praxis der Wirtschaftsinformatik, Heft 227, 2002.

[SeWu02] Seisreiner A., Wurster M.: *Restrukturierungsmanagement im Spannungsfeld von Effizienz, Effektivität und Koordination.* General Management Institute Potsdam e.V., Potsdam, 2002.

[Stör00] Störrle H.: *Models of Software Architecture. Design and Analysis with UML and Petri-nets.* Dissertation, Ludwig-Maximilians-Universität München, 2000.

[Stör01] Störrle H.: *Describing Process Patterns with UML*. LNCS 2077 Software Process Technology, Springer, 2001, pp. 173-181.

[UML1.5] Object Management Group: *OMG Unified Modeling Language Specification*. Version 1.5. Link: http://www.omg.org/docs/formal/03-03-01.pdf (Zugriff am 06.11.2007).

[UML2] Object Management Group: *OMG Unified Modeling Language.* Version 2.1.2. Link: http://www.omg.org/docs/formal/07-11-04.pdf (Zugriff am 22.02.2008).

[Wess96] Wess S.: *Fallbasiertes Problemlösen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik.* Band 126 der Reihe *Dissertationen zur künstlichen Intelligenz*. Infix Verlag, St. Augustin, 1996.