Universität Leipzig
Fakultät für Mathematik und Informatik
Institut für Informatik

Die Modellierung eines Grundbuchsystems im Situationskalkül

Diplomarbeit

vorgelegt von
Steffen Bittner

betreut von
Prof. Dr. Andreas Frank (Technische Universität Wien)
Prof. Dr. Gerhard Brewka (Universität Leipzig)

Leipzig, September 1998

Danksagung

An dieser Stelle möchte ich die Gelegenheit nutzen, um all denen zu danken, die mir den Abschluß dieser Arbeit ermöglicht haben, sei es, indem sie mir mit Rat und Tat zur Seite standen, mir mit ihrer konstruktiven Kritik weiterhalfen oder mir den Rücken freigehalten haben, damit ich mich ganz meiner Aufgabe widmen konnte.
An erster Stelle möchte ich hier meine Freundin Katrin Echtermeyer nennen, die mir sowohl im täglichen Leben Kraft und Motivation gegeben als auch durch ihre Kritik sehr weitergeholfen hat. In diesem Zusammenhang sei auch allen anderen kritischen Lesern gedankt, insbesondere Jan Kowalski und Ralph Miarka. Letzterem besonders auch für viele Diskussionen, in denen einige Ideen geboren wurden, die später Eingang in diese Arbeit fanden.
Besonderer Dank gilt meinen beiden Betreuern Andreas Frank und Gerhard Brewka. Sie ließen mir einerseits den Freiraum, meine eigenen Vorstellungen umzusetzen, andererseits wachten sie darüber, daß ich auf dem richtigen Weg bleibe. Sie waren für mich jederzeit ansprechbar, wenn sich Probleme vor mir auftürmten. Sie verstanden es, mir in den richtigen Augenblicken die notwendige Motivation zu geben, um den nächsten Schritt bewältigen zu können.
Nicht unerwähnt soll die Hilfe meines Bruders Thomas bleiben, der mich insbesondere auf dem Weg von der ersten Idee hin zu einem realisierbaren Konzept sehr unterstützt hat.
An dieser Stelle möchte ich auch an meine Eltern denken, ohne die ich sicher heute nicht diese Zeilen schreiben würde, haben sie mir doch erst den Weg geebnet, auf dem ich mich heute befinde.

Inhalt

1 Einleitung 7
1.1 Motivation 7
1.1.1 Die Bedeutung von Grund und Boden 7
1.1.2 Kataster und Grundbuch 7
1.1.3 Die Automatisierung des Grundbuches 8
1.1.4 Probleme bei der Repräsentation 8
1.2 Grundlagen der Arbeit 8
1.2.1 Einschränkungen 8
1.2.2 Die Methode 9
1.3 Ziel der Arbeit 9
1.4 Der Aufbau der Arbeit 9
2 Das Grundbuch 10
2.1 Einleitung 10
2.2 Zweck, Aufgaben und Bedeutung des Grundbuchs 10
2.3 Das Verhältnis von formellem Recht, materiellem Recht und Grundbuch 11
2.4 Die Grundsätze des Grundbuchrechts 11
2.5 Die Arbeitsweise des Grundbuchamts 12
2.5.1 Der Aufbau des Grundbuchs 12
2.5.2 Das Grundbuchverfahren 13
2.5.3 Die Eintragung in das Grundbuch 13
2.5.4 Der Antrag 14
2.6 Kataster und Grundbuch 14
2.6.1 Der Grundstücksbegriff 14
2.6.2 Beziehungen zwischen Kataster und Grundbuch 15
2.7 Veränderungen im Grundbuchsystem 15
2.7.1 Elementare Veränderungen 15
2.7.1.1 Rechtliche Veränderungen 15
2.7.1.2 Räumliche Veränderungen 16
2.7.1.3 Beziehungen zwischen Operationen in Grundbuch und Kataster 17
2.7.2 Abläufe 17
2.7.2.1 Der Charakter von Abläufen im Grundbuchsystem 17
2.7.2.2 Klassifikation von Abläufen 17
2.7.3 Äußere Einflüsse 18
2.8 Inkonsistenzen 18
2.8.1 Inkonsistenz und Grundbuchunrichtigkeit 18
2.8.2 Arten der Grundbuchunrichtigkeit 19
2.8.3 Ursprüngliche Grundbuchunrichtigkeit 19
2.8.4 Nachträgliche Grundbuchunrichtigkeit 19
2.8.5 Die Ursachen der Grundbuchunrichtigkeit 20
2.8.6 Die Korrektur des unrichtigen Grundbuchs 21
2.9 Zusammenfassung 22
3 Das konzeptionelle Modell 23
3.1 Einleitung 23
3.2 Konzeptionelles Modell, Grundbuch und materielles Recht 23
3.3 Der Inhalt des Modells 24
3.4 Das Modell der materiellen Rechtsverhältnisse 24
3.4.1 Objekte 25
3.4.2 Relationen 25
3.4.3 Aktionen 26
3.5 Probleme bei der Repräsentation des Grundbuchs 26

3.5.1 Objekte, Bezeichner und Identitäten 26
3.5.2 Inkonsistenzen durch Abläufe 27
3.5.3 Die Unvollständigkeit der Repräsentation 27
3.5.4 Fehlfunktion des Systems 27
3.6 Das Modell des Grundbuchs 28
3.6.1 Objekte 28
3.6.2 Relationen 28
3.6.3 Elementare Operationen 29
3.6.4 Komplexe Abläufe 31
3.6.4.1 Verfahrensvoraussetzungen 32
3.6.4.2 Grundbuchverfahren 32
3.7 Die Beziehung von materiellem Recht und Grundbuch 33
3.8 Die Berichtigung des unrichtigen Grundbuchs 33
3.9 Zusammenfassung 34
4 Formale Grundlagen 35
4.1 Einleitung 35
4.2 Wissensrepräsentation 35
4.2.1 Anforderungen an die Wissensrepräsentationssprache 36
4.2.2 Warum Prädikatenlogik? 36
4.2.3 Warum Situationskalkül? 36
4.2.4 Reiters Erweiterung des Situationskalküls 37
4.3 Spezifikation 38
4.3.1 Formale Spezifikation und Wissensrepräsentation 38
4.3.2 Spezifikation von Datenbanktransaktionen 38
4.3.3 Spezifikation von komplexen Transaktionen 39
4.3.4 Ausführbare Spezifikationen und Implementation 39
4.4 Formale Grundlagen zur Repräsentation eines Grundbuchsystems - einleitende Gedanken 40
4.5 Die Formalisierung elementarer Transaktionen 40
4.5.1 Der Situationskalkül 40
4.5.2 Beschreibung von Datenbanktransaktionen 41
4.5.3 Die Gewinnung der Updatespezifikation 43
4.5.4 Das Frame Problem 44
4.5.5 Wie sieht die Datenbank aus? 45
4.5.6 Der Regressionsoperator R 45
4.6 Die Formalisierung komplexer Transaktionen 47
4.6.1 Prozesse im Situationskalkül 47
4.6.2 Komplexe Transaktionen: GOLOG 48
4.6.3 Repräsentation von Zeit 52
4.6.4 GOLOG mit externen Transaktionen und Reaktionen 52
4.7 Anfragen an das System 55
4.8 Diskussion und Zusammenfassung 55
4.8.1 Warum Logik zur Spezifikation von Datenbanken? 55
4.8.2 Warum ist eine Lösung des Frame Problems notwendig? 56
4.8.3 Angabe der Effektaxiome vs. Angabe der Updatespezifikation 56
4.8.4 Das Beweisen von Eigenschaften der Spezifikation 57
4.8.5 Gelöste und ungelöste Probleme des Situationskalküls 57
5 Das formale Modell 58
5.1 Einleitung 58
5.2 Das Modell der materiellen Rechtsverhältnisse 58
5.2.1 Objekte 58
5.2.2 Relationen 59
5.2.3 Aktionen 59

5.2.3.1 Vorbedingungen für Aktionen: Transaction Precondition Axiome 60
5.2.3.2 Wirkungen von Aktionen: Successor State Axiome 61
5.2.3.3 Die initiale Datenbank 62
5.3 Das Modell des Grundbuchs 62
5.3.1 Objekte 63
5.3.2 Relationen 63
5.3.3 Die Beschreibung der elementaren Operationen 63
5.3.3.1 Vorbedingungen für Transaktionen: Transaction Precondition Axiome 65
5.3.3.2 Die Wirkungen von Transaktionen: Successor State Axiome 65
5.3.3.3 Die initiale Datenbank 66
5.3.4 Die Beschreibung der komplexen Abläufe 66
5.3.4.1 Anträge 67
5.3.4.2 Der Eintragungsprozeß 67
5.3.4.3 Die Grundbuchverfahren 68
5.4 Das Modell der Beziehungen zwischen materiellem Recht und Grundbuch 71
5.4.1 Konsistenzbedingungen 71
5.4.2 Inkonsistenzmarkierungen 71
5.4.3 Aussagen über das Verhältnis von materieller Rechtslage und Grundbuch 72
5.4.4 Die Korrektur des unrichtigen Grundbuchs 73
5.4.4.1 Beschwerden 73
5.4.4.2 Verfeinerung der Inkonsistenzmarkierungen 74
5.4.4.3 Beschreibung des korrigierenden Ablaufs 74
5.5 Zusammenfassung 75
6 Implementation 76
6.1 Einleitung 76
6.2 Formale Grundlagen der Implementation 76
6.2.1 Begriffe und Schreibweisen 76
6.2.2 Definitionstheorien und Prolog 77
6.2.3 Die Überführung von Formeln in Klausel Form 78
6.3 Die Implementation elementarer Transaktionen 79
6.3.1 Transaction Precondition Axiome als Definitionen 79
6.3.2 Successor State Axiome als Definitionen 80
6.3.3 Das Implementationstheorem 81
6.3.4 Regression und Prolog 82
6.4 Die Implementation komplexer Transaktionen 82
6.4.1 Der GOLOG Interpreter 82
6.4.2 Der sequentielle temporale GOLOG Interpreter 83
6.4.3 Der RGOLOG Interpreter 84
6.4.4 Der GOLOG Interpreter zur Implementation eines Grundbuchsystems 84
6.5 Die Implementation des Grundbuchsystems 85
6.5.1 Grundgedanken 85
6.5.2 Implementation des Zeitbegriffs 85
6.5.3 Die Visualisierung der Daten 86
6.5.4 Das Hauptprogramm 87
6.5.5 Die Implementation der materiellen Rechtsverhältnisse 88
6.5.6 Die Implementation des Grundbuchs 88
6.5.7 Die Implementation der Inkonsistenzmarkierungen 89
6.5.8 Implementation der Korrektur des unrichtigen Grundbuchs 90
6.5.9 Das Grundbuch als Auskunftssystem 90
6.6 Der Test der Implementation 90
6.6.1 Das Testszenario 91
6.6.2 Der normale Ablauf 92
6.6.3 Die Markierung von Inkonsistenzen 93

6.6.4 Die Korrektur des unrichtigen Grundbuchs 96
6.7 Zusammenfassung 98
7 Zusammenfassung 99
7.1 Was wurde getan? 99
7.2 Ergebnisse 99
7.2.1 Räumliche und zeitliche Veränderungen 99
7.2.2 Grundbuch und materielle Rechtsverhältnisse 99
7.2.3 Wissensrepräsentation und Spezifikation 100
7.2.4 Das Modell des Grundbuches als Datenbankspezifikation 100
7.3 Was bleibt zu tun? 100
7.4 Schlußfolgerungen 100
8 Literatur 99

1 Einleitung

1.1 Motivation

> *„Grund und Boden ist nicht beliebig vermehrbar, er ist als nahezu einziges Gut unverzichtbar und unzerstörbar. Er genießt daher besondere Wertschätzung und sein Wert nimmt zu, ...“ (Eickmann 1986, S.1)*

1.1.1 Die Bedeutung von Grund und Boden

Grundlage jeder menschlichen Besiedlung ist der Grund und Boden, ohne den somit menschliche Existenz nicht möglich wäre. Boden bildet die Grundlage der Ernährung des Menschen. Wie sollte sonst ein Feld entstehen, auf dem Getreide wächst? Wo sollten Menschen ihre Siedlungen bauen, wenn nicht auf dem Grund und Boden, der ihr Eigentum ist? Er ist so selbstverständlich und allgegenwärtig, daß seine Wichtigkeit leicht in den Hintergrund tritt. Deshalb wird diese fundamentale Bedeutung an dieser Stelle gewürdigt. Viel bekannter und bewußter ist dem Menschen der finanzielle Gegenwert von Grund und Boden, der eine stabile und vor Währungsturbulenzen sichere Geldanlage oder einfach nur die finanzielle Sicherheit im Alter bedeutet.
In ihrer Entwicklung hat die Menschheit ihren Lebensraum immer weiter ausgedehnt und stieß in die entlegensten Gebiete vor. Dabei offenbarte sich ein weiterer Aspekt. Grund und Boden ist nicht vermehrbar, er ist eine beschränkte Ressource, die sich immer mehr Menschen teilen. Auf Grund seines finanziellen Gegenwerts ist der Grund und Boden zu einem gewaltigen Wirtschaftsfaktor geworden. Er bildet die Grundlage für den Realkredit, also der hypothekarischen Beleihung des Bodens. Viele Investitionen, wie beispielsweise der Bau von Häusern, wären nicht möglich, könnten nicht die finanziellen Mittel durch die Beleihung des Bodens freigesetzt werden.
Die Bedeutung einerseits und die beschränkte Verfügbarkeit andererseits begründen den herausragenden Stellenwert von Grund und Boden in unserer Gesellschaft. Um diesem Stellenwert gerecht zu werden, ist eine korrekte und eindeutige Verwaltung der Ressource notwendig. Zur Gewährleistung des friedlichen Zusammenlebens der Menschen ist es von großer Bedeutung, Streitigkeiten über Grund und Boden zu vermeiden. Dazu müssen die Eigentumsverhältnisse klar und für jeden ersichtlich offenliegen. Es werden deshalb große Anstrengungen unternommen, um den Grund und Boden so zu verwalten, daß er diesen Ansprüchen gerecht werden kann.
International entwickelten sich verschiedene Lösungen zur Verwaltung von Grund und Boden, die unter den Begriffen Kataster und Grundbuch zusammengefaßt werden.

1.1.2 Kataster und Grundbuch

Zu einer konsistenten Verwaltung von Grund und Boden ist es notwendig, zwei Aufgaben zu lösen. Einerseits müssen die Landstücke, die den Boden unterteilen, entsprechend ihrer Form und Lage eindeutig beschrieben und andererseits den Personen, die Rechte an diesen Grundstücken besitzen, zugeordnet werden. Die erste Aufgabe ist dem Kataster zugeordnet, die zweite dem Grundbuch. Der zentrale Unterschied dieser Einrichtungen liegt darin, daß das Kataster Teile der Erdoberfläche entsprechend ihrer räumlichen Lage und Ausdehnung verwaltet. Grundbücher hingegen wurden geschaffen, um über Rechtsverhältnisse an Grund und Boden Aufschluß zu geben (Löffler 1993, S.1).
Die Bedeutung von Grundbuch und Kataster ergibt sich aus der Bedeutung des Grund und Bodens selbst. Sie sind die Voraussetzung, daß der Grund und Boden entsprechend seiner Bedeutung genutzt und über seinen Wert verfügt werden kann.

1.1.3 Die Automatisierung des Grundbuchs

Grundbücher werden auf Grund der Bedeutung des Bodens in der ganzen Welt mit hohem Aufwand gepflegt. Es besteht daher ein großer Bedarf, die Funktionen des Grundbuchs effektiv zu gestalten. Mit der Entwicklung der Computertechnik in den letzten Jahrzehnten steht nun ein Werkzeug bereit, das genau diesen Bedarf decken kann. Computer ermöglichen es einerseits, große Datenmengen zu verwalten und effektiv verfügbar zu machen, und andererseits die Korrektheit dieser Daten zu erhalten. Zu genau diesem Zweck, die Verwaltung großer Datenmengen zu realisieren, wurden Datenbanksysteme geschaffen. Geographische Informationssysteme (GIS) (Laurini und Thompson 1992) wurden in den letzten Jahren entwickelt, um räumliche Daten zu verwalten. Entsprechend der Bedeutung von Grund und Boden geht die weitere Entwicklung auch in die Richtung, Kataster und Grundbuch in GIS-Systeme zu integrieren. Angestrebt wird die Erstellung eines sogenannten *Ownership Layers* (Frank 1996), das diese Aufgaben erfüllen kann.

1.1.4 Probleme bei der Repräsentation

Bei der Repräsentation eines Systems zur Verwaltung von Grund und Boden treten verschiedene Probleme auf, deren genaue Analyse Grundlage für eine erfolgreiche Implementation ist.
Große Bedeutung kommt der Betrachtung von Veränderungen des Systems zu. Konventionelle Datenbanksysteme sind stark darauf ausgerichtet, eine Menge von Daten statisch zu verwalten. Veränderungen in einem solchen System sind lediglich Übergänge zwischen den konsistenten Zuständen des Systems. Eine solche Sichtweise trägt nicht der Tatsache Rechnung, daß sich Systeme in ständiger Veränderung befinden, diese Veränderungen also fundamentale Bestandteile dieser Systeme sind. Die Betrachtung von Veränderungen als Übergang zwischen statischen Zuständen stellt lediglich eine idealisierende Vereinfachung dar und ist nicht geeignet, alle notwendigen Aspekte zu beschreiben.
Problematisch bei der Beschreibung eines Grundbuchsystems ist, daß jede Repräsentation Grenzen hat. Das heißt, daß die Repräsentation unvollständig bleiben muß, da nicht alle relevanten Aspekte bekannt sind oder die Komplexität des Systems beschränkt werden muß. Es ergeben sich große Probleme daraus, daß die Repräsentation von Rechtsverhältnissen im Grundbuch und die Rechtsverhältnisse selbst nicht übereinstimmen, obwohl dies das erklärte Ziel eines solchen Systems ist. Hieraus erwächst auch eine große Bedeutung der Repräsentation eines Grundbuchsystems, wenn es in der Lage ist, diese Nichtübereinstimmung gering und kontrolliert zu halten.

1.2 Grundlagen der Arbeit

1.2.1 Einschränkungen

Hauptproblem jeder wissenschaftlichen Arbeit ist es, das zu beschreibende Problem so zu formulieren, daß es auch lösbar ist. Dazu ist es notwendig, bestimmte Einschränkungen zu treffen. In dieser Arbeit gelten deshalb die folgenden Randbedingungen:

- Grundlage der Beschreibung ist das deutsche Recht. Die Analyse des Grundbuches bezieht sich auf die deutsche Grundbuchordnung (GBO 1987-1995), im weiteren GBO, und das Bürgerliche Gesetzbuch (BGB 1987), im weiteren BGB. Dadurch ergibt sich auf der einen Seite, daß verschiedene Aspekte nicht auf andere Länder zu übertragen sind, und daß auf der anderen Seite Probleme unbehandelt bleiben. Dies liegt daran, daß sich in den verschiedenen Ländern unterschiedliche Systeme entwickelt haben, die im Detail sehr stark voneinander abweichen. Es gilt jedoch, daß die grundlegenden Zusammenhänge auf Grund der gleichen Zielstellung auch gleich sind, und deshalb Ergebnisse dieser Arbeit auch zu verallgemeinern sind.
- Diese Arbeit beschreibt das Grundbuchsystem, klammert also das Kataster weitgehend aus, obwohl beide in engem Zusammenhang stehen. Betrachtet werden Veränderungen in

den Rechtsverhältnissen und in der rechtlichen Identität von Grundstücken, nicht aber ihre räumliche Lage oder Ausdehnung.

1.2.2 Die Methode

Wichtig für diese Arbeit ist das Vorgehen, also die Methode, die angewandt wird, um die Probleme bei der Repräsentation eines Grundbuchsystems zu lösen.
Grundsätzlich orientiert sich diese Arbeit an dem deklarativen Ansatz (Russell und Norvig 1995), um das Wissen über das System klar und strukturiert repräsentieren zu können. Als Repräsentationssprache wird die Prädikatenlogik verwandt, die im Bereich der Wissensrepräsentation sehr verbreitet ist. Der konkrete Kalkül, der zur Beschreibung des Systems und damit zur Erstellung eines Modells genutzt wird, ist der Situationskalkül von John McCarthy (McCarthy und Hayes 1969).

1.3 Ziel der Arbeit

Die Arbeit zielt auf die Erstellung eines ausführbaren Prototypen eines Grundbuchsystems auf Basis des erstellten Modells. Anhand von verschiedenen Fallbeispielen wird getestet, ob der Prototyp das erstellte Modell korrekt implementiert. Entscheidender Schwerpunkt ist die Beschreibung der Veränderungen im Grundbuchsystem unter der Bedingung, daß die Repräsentation unvollständig bleibt und deshalb Inkonsistenzen auftreten.

1.4 Der Aufbau der Arbeit

Kapitel 2 ist der Analyse des Grundbuchsystems gewidmet. Aufbauend auf diese Analyse wird in Kapitel 3 ein konzeptionelles Modell entworfen, das informell die wesentlichen Aspekte des Grundbuchsystems beschreibt. Aus den Anforderungen des Modells ergeben sich die notwendigen formalen Grundlagen, um das Modell abzubilden. Kapitel 4 beschäftigt sich mit der Beschreibung der formalen Grundlagen. Dabei wird insbesondere auf den Zusammenhang zwischen Wissensrepräsentation und formaler Spezifikation eingegangen, der die Verwendung des Situationskalküls zur Spezifikation eines Grundbuchsystems begründet. In Kapitel 5 erfolgt die Formalisierung des konzeptionellen Modells mit Hilfe des in Kapitel 4 beschriebenen Ansatzes. Ergebnis ist das formale Modell. Dies ist Grundlage für die Implementation, die in Kapitel 6 vorgestellt wird. Die Implementation erfolgt in Prolog (Bratko 1990). Dazu wird die Korrektheit der Prolog Implementation bezüglich des formalen Modells formal begründet. Wichtiger Punkt ist der Test des erstellten Modells anhand von Beispielen. Im Kapitel 7 werden die gewonnenen Ergebnisse diskutiert.

2 Das Grundbuch

2.1 Einleitung

In diesem Kapitel werden Aufbau und Bedeutung des Grundbuchs erläutert. Hierbei wird von den Aufgaben und Anforderungen ausgegangen, die das Grundbuch erfüllen soll. Fundamental ist im deutschen Recht die Unterteilung in materielles und formelles Recht. Dies stellt die Grundlage für die Formulierung der Grundsätze dar, auf denen das Grundbuchrecht basiert, damit es seinen Anforderungen gerecht werden kann.
Im nachfolgenden Abschnitt wird geschildert, wie sich diese Grundsätze in der Arbeitsweise des Grundbuchamts niederschlagen, wie sie also umgesetzt werden. Sehr eng mit dem Grundbuch ist das Kataster verbunden. Deshalb kann die Arbeitsweise des Grundbuchamts nicht beschrieben werden, ohne diese Zusammenhänge näher zu untersuchen. Von zentraler Bedeutung für das Grundbuchrecht sind die Regeln, nach denen das Grundbuch sich verändert. Insbesondere muß die Richtigkeit des Grundbuchs erhalten werden unter der Bedingung, daß ständig Veränderungen stattfinden. Im Vordergrund steht deshalb die Beschreibung dieser Veränderungen und ist elementare Grundlage der Modellierung des Grundbuchsystems, die Inhalt dieser Arbeit ist. Um die Richtigkeit des Grundbuchs zu erhalten, ist es wichtig, die Ursachen von Inkonsistenzen zu kennen. Deshalb werden die Inkonsistenzen als Abschluß dieses Kapitels näher betrachtet.

2.2 Zweck, Aufgaben und Bedeutung des Grundbuchs

Grund und Boden genießt in unserer Gesellschaft einen hohen Stellenwert. Dies liegt einerseits daran, daß er Grundlage jeder Besiedlung durch den Menschen und andererseits nicht vermehrbar ist. Deshalb ist es notwendig, die Rechtsverhältnisse an Grund und Boden eindeutig klarzustellen. Insbesondere ist es wichtig zu bestimmen, wer der Eigentümer eines Grundstücks ist und welche Rechte an einem Grundstück bestehen. Hierzu wurde das Grundbuch geschaffen. Das Zusammenleben der Menschen auf dem Grund und Boden und der Wert, den er darstellt, bergen naturgemäß die Gefahr von Interessengegensätzen und somit ein hohes Konfliktpotential in sich. Deshalb ist es oberstes Prinzip bei der Einrichtung des Grundbuchs, Rechtssicherheit zu schaffen. Diese Rechtssicherheit wird dadurch erreicht, daß einerseits der Inhalt des Grundbuchs, also die Eigentumsverhältnisse, öffentlich gemacht werden und andererseits für die Richtigkeit des Grundbuchs garantiert wird. Öffentlich wird das Grundbuch dadurch gemacht, daß ein jeder, der ein berechtigtes Interesse nachweisen kann, in das Grundbuch Einsicht nehmen darf. Die Garantie der Richtigkeit des Grundbuchs bewirkt, daß jeder, der ein im Grundbuch eingetragenes Recht erwirbt, auch Eigentümer dieses Rechts wird, selbst wenn das Grundbuch nicht die wirkliche Rechtslage wiedergab. Dies bewirkt, daß der Staat auch für eventuelle aus dieser Unrichtigkeit erwachsene Schadenersatzansprüche einsteht.
Neben der Offenlegung der Eigentumsverhältnisse an sich müssen die Verhältnisse der Rechte untereinander, also ihr Rang festgelegt werden. Dadurch wird erreicht, daß wann immer es zu einem Konflikt zwischen Ansprüchen kommt, die dasselbe Recht betreffen, das vorrangige Recht die Priorität besitzt.
Die große wirtschaftliche Bedeutung des Grundeigentums liegt darin, daß es Grundlage für den Realkredit ist. Hierunter wird die hypothekarische Beleihung des Bodens verstanden. Möglich wurde dies durch die Schaffung des Grundbuchs mit der damit garantierten Rechtssicherheit. Neben den Eigentumsverhältnissen müssen daher die Belastungen an einem Grundstück eingetragen werden.
Das Grundbuch ist also ein öffentliches Register, das über die Rechtsverhältnisse an den Grundstücken Auskunft gibt (Löffler 1993, S.11). Hierbei muß beachtet werden, daß lediglich die privatrechtlichen Verhältnisse an Grundstücken, nicht aber die öffentlich-rechtlichen Verhältnisse Eingang in das Grundbuch finden (Demharter 1997, S.1).

Zusammenfassend bestehen die Aufgaben des Grundbuchsystems in:

- der Offenlegung der Eigentumsverhältnisse an Grundstücken,
- der Kenntlichmachung von dinglichen Belastungen, sowie
- der Feststellung ihrer Rangverhältnise untereinander (Eickmann 1986, S.2).

2.3 Das Verhältnis von formellem Recht, materiellem Recht und Grundbuch

Das gesamte deutsche Recht ist von der Zweiteilung in materielles und formelles Recht geprägt. Hierbei regelt das materielle Recht die Beziehungen zwischen Menschen und die Zuordnungen zwischen Menschen und Sachen. Das formelle Recht legt die Normen und Verfahrensweisen fest, die der Klarstellung und Durchsetzung dieser materiellen Rechte dienen. Es wird auch als Verfahrensrecht bezeichnet (Bengel und Simmerding 1995, S.132). Aufgabe des Grundbuchs ist es, die materiellen Rechte an Grundstücken offenzulegen. Damit es dieser Aufgabe gerecht werden kann, muß es auch die verfahrensrechlichen Normen abbilden, die für die Umsetzung des materiellen Rechts notwendig sind. Das Grundbuch bildet folglich nicht nur die materiellen Rechtsverhältnisse ab, sondern auch das Verfahrensrecht.

2.4 Die Grundsätze des Grundbuchrechts

Damit das Grundbuch seinen Aufgaben gerecht werden kann, liegen ihm bestimmte Prinzipien zugrunde, die im folgenden erläutert werden (Bengel und Simmerding 1995, S.136ff).
Dies sind im einzelnen:

- Der Einigungsgrundsatz (Konsensprinzip)

Jede auf einem Rechtsgeschäft beruhende Rechtsveränderung an einem Grundstück bedarf, um gültig zu werden, zweier Bestandteile: der Einigung des bisher Berechtigten und des künftig Berechtigten über den Eintritt der Änderung und der Eintragung im Grundbuch. Diese Bestandteile bewirken die Rechtsänderung. Dieses Prinzip wird als das *materielle Konsensprinzip* bezeichnet. Das Grundbuchrecht weicht von diesem Prinzip ab, an seine Stelle tritt das *formelle Konsensprinzip*. Es besagt, daß als Grundlage für die Eintragung im Grundbuch nur eine einseitige Erklärung desjenigen notwendig ist, dessen Recht von der Eintragung betroffen (also eingeschränkt) wird (§19 GBO). Diese Erklärung wird als Bewilligung bezeichnet.

- Der Eintragungsgrundsatz (Buchungsprinzip)

Das *materielle Buchungsprinzip* besagt, daß die Eintragung eines Rechts im Grundbuch konstituive Wirkung hat, also neues Recht schafft. Ohne diese Eintragung tritt die Rechtsänderung nicht ein. Zusammen mit dem Konsensprinzip heißt das, daß Eintragung und Einigung untrennbar verbunden sind, also nur durch beides zusammen neues Recht erzeugt wird.

- Der Antragsgrundsatz

Das Grundbuchamt soll nur auf Antrag tätig werden. Das heißt, daß Voraussetzung für die Erreichung einer Eintragung im Grundbuch das Stellen eines Antrags beim Grundbuchamt ist. Von diesem Prinzip gibt es Ausnahmen. Insbesondere bei Bekanntwerden einer Unrichtigkeit im Grundbuch hat das Grundbuchamt auch ohne Antrag tätig zu werden und einen Amtswiderspruch einzutragen. Ein anderer Bereich ist die Zusammenarbeit mit dem Katasteramt zur Erhaltung der Übereinstimmung zwischen Grundbuch und Kataster.

- Der Prüfungsgrundsatz (Legalitätsprinzip)

Das Grundbuchamt hat das Recht und die Pflicht, eingehende Anträge zu prüfen. Dieses Prinzip nennt man *Legalitätsprinzip*. Es verhindert, daß das Grundbuch unrichtig wird. Dieses Prinzip erstreckt sich lediglich auf das formelle Grundbuchrecht, etwa auf die Prüfung der Antragsberechtigung. Es erstreckt sich nicht auf das materielle Recht, etwa darauf, ob ein

Recht auch wirklich übergegangen ist. Seine Ausprägung findet dieser Grundsatz auch in der Forderung nach dem Nachweis der Eintragungsunterlagen in der Form beglaubigter Urkunden.

- Der Öffentlichkeitsgrundsatz (Publizitätsprinzip)

Der Inhalt des Grundbuchs gilt zu Gunsten einer Person, die ein Recht erwirbt, als richtig, sofern der Person nicht bekannt ist, daß es unrichtig ist. Dieses Prinzip begründet den öffentlichen Glauben des Grundbuchs und verwirklicht das *materielle Publizitätsprinzip*. Das bedeutet, daß der Inhalt des Grundbuchs zu Gunsten eines gutgläubigen Erwerbers grundsätzlich als richtig gilt. Das *formelle Publizitätsprinzip* besagt, daß derjenige, der ein berechtigtes Interesse hat, Einsicht in das Grundbuch nehmen darf. Diejenigen Personen, die von der Eintragung eines Rechts betroffen sind, müssen benachrichtigt werden.

- Der Bestimmtheitsgrundsatz (Spezialitätsprinzip)

Dieses Prinzip besagt, daß der Gegenstand des Rechtsverkehrs im Grundbuch klar und eindeutig erkennbar sein muß. Daraus folgt, daß in das Grundbuch alles gehört, was zur Feststellung eines Rechts und seines Umfangs notwendig ist. Insbesondere muß das Grundstück, das Gegenstand des Rechtsverkehrs ist, eindeutig bestimmt sein. In diesem Zusammenhang ist auch die Übereinstimmung mit den entsprechenden Katastereintragungen zwingend notwendig, bilden diese doch die Grundlage für die Darstellung der Grundstücke im Grundbuch.

- Der Vorranggrundsatz (Prioritätsprinzip)

Das Prioritätsprinzip besagt, daß bei Eintragungen, die im Konflikt zueinander stehen, die Eintragungsreihenfolge entscheidet. Bei Eintragungen an verschiedenen Stellen erhält das Recht den höheren Rang, dessen Eintragsdatum älter ist.

2.5 Die Arbeitsweise des Grundbuchamts

Der folgende Abschnitt erläutert die Grundregeln der Arbeitsweise des Grundbuchamts. Dabei wird insbesondere beachtet, wie die Grundsätze des Grundbuchrechts Eingang in die Arbeit des Amts finden.

2.5.1 Der Aufbau des Grundbuchs

Das Grundbuch besteht aus Grundbuchbänden, in denen die einzelnen Grundbuchblätter gelagert werden. Grundbuch im Rechtssinn ist nicht der Grundbuchband, sondern das Grundbuchblatt, auf dem das Grundbuch gebucht ist. Für jedes Grundstück wird ein solches Blatt geführt, das sogenannte *Realfolium*. Von diesem Grundsatz weicht man manchmal ab. Dabei werden mehrere, einer Person gehörende Grundstücke auf einem Blatt gebucht, dem *Personalfolium*. Dies dient der besseren Übersichtlichkeit des Grundbuchs. Über den Aufbau eines Grundbuchblatts gibt es klare Regeln, die hier nicht weiter erläutert werden. Es sei verwiesen auf (Bengel und Simmerding 1995).

Neben dem eigentlichen Grundbuch führt das Grundbuchamt weitere Unterlagen. Dazu gehören die Grundakten. In den Grundakten lagern die zum Beweis eines Rechts erbrachten Urkunden. Das Grundbuchamt ist berechtigt, neben den Grundakten weitere Verzeichnisse zu führen, die für seine Arbeit notwendig sind. Beispielsweise das Eigentümerverzeichnis, das Namen und Adressen der Grundstückseigentümer enthält.

Die Unterlagen des Grundbuchsystems können in zwei Kategorien eingeteilt werden. Das sind einerseits die Unterlagen, die über die materiellen Rechtsverhältnisse selbst Auskunft geben, also das eigentliche Grundbuch. Zur zweiten Kategorie gehören die Unterlagen, die die Rechtsverhältnisse und ihre Veränderungen dokumentieren bzw. das Zustandekommen der Rechtsverhältnisse begründen. Hierunter fallen die oben genannten Grundakten und alle weiteren Verzeichnisse. Diese zweite Kategorie bildet das formelle Grundbuchrecht ab.

2.5.2 Das Grundbuchverfahren

Charakterisiert ist die Arbeitsweise des Grundbuchsystems durch das Grundbuchverfahren. Innerhalb des Grundbuchverfahrens werden alle Entscheidungen des Amts getroffen. Das heißt, daß alle Veränderungen in einem Grundbuch innerhalb eines solchen Verfahrens erfolgen. Jeder Grundbucheintrag ist Ergebnis eines Grundbuchverfahrens. Andersherum zielt jedes Grundbuchverfahren auf die Erreichung einer bestimmten Eintragung ab. Es endet mit der Entscheidung, ob eine Eintragung vorzunehmen ist oder nicht. Entsprechend dem Legalitätsprinzip prüft das Amt, ob die Voraussetzungen für eine Eintragung gegeben sind oder nicht. In Abhängigkeit dieser Prüfung trifft es seine Entscheidung.
Entsprechend dem Antragsprinzip wird das Grundbuchamt tätig nach Eingang eines Antrags auf Eintragung eines Rechts. Damit wird das Grundbuchverfahren in Gang gesetzt. Nachdem das Grundbuchamt die Verfahrensvoraussetzungen, zum Beispiel die Zuständigkeit, geprüft hat, erfolgt die Bearbeitung des Antrags. Das Grundbuchamt entscheidet dann nach Würdigung der Beweise, die zum Nachweis dieses Rechts erbracht wurden, ob ein Antrag angenommen oder abgelehnt wird. Als Beweise gelten dem Amt vorgelegte und beglaubigte Dokumente. Die Umsetzung des Legalitätsprinzips erstreckt sich also auf die Einhaltung der Verfahrensregeln, nicht aber auf die Prüfung der Übereinstimmung mit dem materiellen Recht. Um den Grad der Entsprechung möglichst hoch zu halten, gehört es zu den Regeln des Verfahrens, daß nur beglaubigte Urkunden als Beweise gültig sind. Dies zwingt die Beteiligten dazu, ihre Interessen klar und deutlich zu formulieren und engt den Spielraum für Betrug ein. Die Hinzuziehung einer unbeteiligten, dritten Person, die die Urkunde beglaubigt, vermeidet Probleme, die sich aus Mißverständnissen und Unkenntnis ergeben.
Das Grundbuchverfahren kann in vielen Fällen komplexer sein. Nicht immer beschränkt es sich auf die Prüfung der Voraussetzungen für eine Eintragung und ihre Durchführung. Es können komplexere Änderungen im Grundbuch notwendig sein, um das Verfahren abzuschließen. Innerhalb eines Grundbuchverfahrens können verschiedene Teilverfahren ablaufen. Für den Fall, daß der Verkauf eines Grundstücksteils eingetragen werden soll, ist es zum Beispiel notwendig, den zu verkaufenden Teil vorher vom Grundstück abzuteilen. Weiterhin müssen innerhalb des Verfahrens die Beweisdokumente registriert werden. Das Grundbuchsystem bildet das Gesamtverfahren in eine solche Folge von Operationen ab, die auf den Grundbuchdaten stattfinden. Nun sind auf den Grundbuchdaten nicht beliebige Operationen möglich, sondern nur einige bestimmte, wie zum Beispiel die Grundstücksteilung.
Sehr eng mit dem Grundbuch ist das Kataster verbunden. Grundstücke sind im Grundbuch basierend auf den im Kataster verwalteten Flurstücken dargestellt. Operationen, die im Grundbuch ablaufen, können bestimmte Operationen im Kataster erfordern.
Das Grundbuchverfahren ist gekennzeichnet durch die Folge der Operationen, die im Grundbuch ablaufen, also den Eintragungen. Ob und welche Eintragungen vorgenommen werden, wird innerhalb des Verfahrens festgelegt. Die Beweiswürdigung oder die Interpretation von Rechtsvorschriften sind richterliche Entscheidungen. Das Grundbuchverfahren wird also entscheidend geprägt durch die menschlichen Entscheidungen innerhalb des Grundbuchamts.
Zusammenfassend kann gesagt werden, daß zwei Faktoren für das Grundbuchverfahren maßgeblich sind: Einerseits die Operationen im Grundbuch, also die Eintragungen, und andererseits die menschlichen, zum Beispiel richterlichen Entscheidungen.

2.5.3 Die Eintragung in das Grundbuch

Für die Eintragung in das Grundbuch sind drei Voraussetzungen notwendig (Löffler 1993, S.34):

- Der Antrag auf Eintragung,
- die Bewilligung der Eintragung durch den, der durch die Eintragung betroffen ist,
- die Voreintragung des Berechtigten.

Im ersten Punkt findet das Antragsprinzip seine Ausprägung. Der zweite Punkt setzt das formelle Konsensprinzip um, indem die Bewilligung dessen verlangt wird, dessen Recht von der Eintragung eingeschränkt wird. Der dritte Punkt drückt das materielle Publizitätsprinzip aus. Es bewirkt, daß nur die im Grundbuch eingetragene Person über ein Recht verfügen können. Dies ist das einzige Kriterium. Folglich kann die Person auch über das Recht verfügen, wenn der Grundbucheintrag unrichtig ist. Den gutgläubigen Erwerber schützt also der Grundbucheintrag. Decken sich Grundbucheintrag und die Einigung, wird neues Recht geschaffen (Buchungsprinzip).

2.5.4 Der Antrag

Entsprechend dem Antragsprinzip wird das Grundbuchamt erst tätig, wenn ein solcher Antrag gestellt ist. Um die Möglichkeit der Grundbuchunrichtigkeit zu reduzieren, sind die Verfahrensvoraussetzungen zu prüfen. Hierzu ist als erstes zu klären, ob das angerufene Grundbuchamt zuständig ist, also ob das betreffende Grundstück von ihm verwaltet wird. Weiterhin ist zu prüfen, ob die beantragende Person zum Antrag berechtigt ist. Berechtigt sind alle Personen, deren Recht von dem Antrag betroffen ist bzw. die durch die angestrebte Eintragung gewinnen. Im Falle eines Verkaufs wären es also Käufer und Verkäufer. Um das Spezialitätsprinzip umzusetzen, muß geprüft werden, ob sowohl die Art des beantragten Rechts als auch das betreffende Grundstück aus dem Antrag klar erkennbar sind.
In der Art, wie Anträge durch das Grundbuchamt bearbeitet werden, findet das Prioritätsprinzip seine Ausprägung. Es entscheidet sie in der Eingangsreihenfolge. Das heißt, daß eine Eintragung, die früher beantragt wurde, auch vor einer später beantragten eingetragen wird. Damit erhält sie den höherrangigen Platz im Grundbuch.

2.6 Kataster und Grundbuch

Eng verbunden mit dem Grundbuch ist das Kataster. Grundlage für die Darstellung eines Grundstücks im Grundbuch ist stets ein im Kataster verwaltetes Flurstück. Das Grundbuch verwaltet die Rechte an Grundstücken sowie die Veränderungen der Grundstücke selbst in Bezug auf ihre rechtliche Identität und ihren Rechtsstatus. Im Gegensatz dazu ist es Aufgabe des Katasters, Flurstücke entsprechend ihrer Lage, Fläche und Nutzung zu beschreiben. Auch wenn die Abbildung des Katasters nicht Inhalt dieser Arbeit ist, so ist es doch für das Verständnis des Grundbuchs hilfreich, auf Zusammenhänge mit dem Kataster hinzuweisen. Deshalb werden diese Beziehungen im folgenden kurz diskutiert.

2.6.1 Der Grundstücksbegriff

Der Grundstücksbegriff im Kataster weicht von dem im Grundbuch ab. Ein Grundstück im Sinn des Katasters ist ein Teil der Erdoberfläche, der im Kataster eindeutig gekennzeichnet, beschrieben und dargestellt ist (Eickmann 1986, S.44). Er wird als Flurstück bezeichnet. Im Grundbuch dagegen ist ein Grundstück ein Teil der Erdoberfläche, das im Grundbuch unter einer eigenen Nummer verzeichnet ist. Während ein Flurstück durch seine Gestalt, Lage und Beschreibung bestimmt ist, charakterisiert ein Grundstück im Grundbuchsinn (im folgenden nur Grundstück) seine Identität. Folglich werden räumliche Veränderungen im Grundbuch durch Veränderungen in dieser Identität abgebildet. Abschnitt 3.5.1 behandelt den Zusammenhang zwischen Objekten und Identitäten ausführlicher.
Entsprechend dem Spezialitätsprinzip sollen die Objekte des Rechtsverkehrs im Grundbuch eindeutig und klar erkennbar sein. Diesem Grundsatz entspricht es, daß nur ganze Grundstücke im Grundbuch eingetragen werden. Zu dieser Erkennbarkeit gehört auch, daß die Ausmaße, die Grenzen und die Lage eines Grundstücks ersichtlich sind. Deshalb gelten zwischen Flurstücken und Grundstücken folgende Beziehungen: Ein Grundstück besteht aus mindestens einem Flurstück. Besteht es aus genau einem Flurstück, wird es als *Idealgrundstück* bezeichnet, sonst als *zusammengesetztes Grundstück*. Ein Grundstück besteht niemals aus einem Teil eines Flurstücks.

2.6.2 Beziehungen zwischen Kataster und Grundbuch

Veränderungen im Grundstücksbestand eines Grundbuchs können nur im Zusammenhang mit dem zugehörenden Kataster betrachtet werden. Grundstücke im Grundbuch bestehen aus Flurstücken im Kataster. Dieser Zusammenhang wird dadurch abgebildet, daß im Grundbuch zu jedem Grundstück die Nummern der zugehörenden Flurstücke geführt werden. Um den Beziehungen zwischen Grund- und Flurstücken Rechnung zu tragen, erfordern Veränderungen im Grundstücksbestand des Grundbuchs somit sehr oft Veränderungen im Flurstücksbestand des Katasters. Wie werden diese Anforderungen in der Arbeitsweise der Ämter umgesetzt? Wenn eine Veränderung im Grundstücksbestand des Grundbuchs durchgeführt werden soll, etwa eine Grundstücksteilung, dann sind im Falles eines Idealgrundstücks zwei neue Flurstücke zu bilden. Dies trägt der Forderung Rechnung, daß jedes Grundstück aus mindestens einem Flurstück bestehen soll. Über diese Veränderung im Kataster führt das Katasteramt einen Veränderungsnachweis, der dem Grundbuchamt übersandt wird. Daraufhin nimmt das Grundbuchamt die Veränderung in seinem Grundstücksbestand vor. Die Zusammenarbeit zwischen Grundbuch- und Katasteramt gestaltet sich nun so, daß das Grundbuchamt, wenn es notwendig ist, eine Veränderung im Kataster beauftragt. Das Ergebnis dieser Veränderung wird dann dem Grundbuchamt als Veränderungsnachweis übersandt.

2.7 Veränderungen im Grundbuchsystem

Veränderungen in einem Grundbuchsystem können entsprechend seiner Arbeitsweise auf zwei Ebenen betrachtet werden. Dies ist einerseits die Ebene des Grundbuchverfahrens und andererseits die Ebene der Operationen auf den Grundbuchdaten. Die Veränderung in Folge eines Grundbuchverfahrens ist die Gesamtheit der Veränderungen durch die Operationen, die innerhalb des Verfahrens ablaufen. In diesem Sinne sind die Operationen elementare Bestandteile eines komplexen Ablaufs, des Grundbuchverfahrens. Die beiden Ebenen werden im folgenden genauer charakterisiert. Dementsprechend werden unterschieden: Einerseits die *elementaren Veränderungen* und die sie verursachenden *elementaren Operationen* und andererseits *komplexe Veränderungen,* verursacht durch *Abläufe.*

2.7.1 Elementare Veränderungen

Das Grundbuch verwaltet Grundstücke und Rechte an Grundstücken. Folglich können Veränderungen, die im Grundbuch abgebildet werden, die Grundstücke selbst oder die Rechte an ihnen betreffen. Dies spiegelt die folgende Einteilung wider.
Veränderungen können in zwei Kategorien eingeteilt werden (Al-Taha 1992):

1. rechtliche Veränderungen: Dazu zählen alle Veränderungen an den Eigentumsverhältnissen. Hierzu gehören der Wechsel des Eigentümers, also der Transfer aller Rechte an einem Grundstück, sowie die Übertragung von Teilrechten, wie zum Beispiel Hypotheken oder Nutzungsrechte wie ein Fischereirecht.
2. räumliche Veränderungen: Nicht nur ihr rechtlicher Status, auch die Gestalt von Grundstücken ändert sich. Grundstücke können geteilt, zusammengeführt oder einem anderen Grundstück zugeschrieben werden.

Im folgenden werden diese Veränderungen zusammen mit den sie verursachenden Operationen vorgestellt.

2.7.1.1 Rechtliche Veränderungen

Rechtliche Veränderungen in einem Grundbuch entstehen, wenn Rechte zwischen Personen transferiert werden. Bei diesen Rechten handelt es sich um Rechte an Grundstücken. Hierbei muß zwischen vollen dinglichen Rechten und beschränkten dinglichen Rechten unterschieden werden (Löffler 1993, S.46). Das Vollrecht ist das Eigentum an einem Grundstück. Das beschränkte Recht gewährt eine begrenzte Herrschaft über eine fremde Sache, also ein

Teilrecht über diese Sache. Gleichzeitig schränkt sie den Eigentümer in seiner Herrschaft über die Sache ein. Als Metapher kann folgendes dienen (Frank 1996): Das Vollrecht ist ein Bündel von Teilrechten. Aus diesem Bündel können Teilrechte abgegeben werden, etwa eine Hypothek wird aufgenommen, oder es kann das gesamte Bündel weitergegeben werden, das Grundstück kann verkauft werden. Ein weiteres Beispiel für ein Teilrecht ist zum Beispiel ein Wegerecht, das eventuell eine Person seinem Nachbarn einräumt, damit dieser sein Grundstück betreten kann. Die wirtschaftliche Wirkung dieser beiden Beispiele ist unterschiedlich. Während eine Hypothek den Verkaufswert einschränkt, beeinträchtigt ein Wegerecht den Gebrauchswert eines Grundstücks. Gegenüber einem Erwerber bleiben diese Rechte bestehen, so sie aus dem Grundbuch ersichtlich sind. Ein anderes Recht, das den Nutzungswert einschränkt, ist der Nießbrauch (Löffler 1993, S.65). Der Nießbrauch ist das Recht, eine Sache zu besitzen und daraus Nutzen zu ziehen (§1036 BGB). Einen Nießbrauch bestellt man sehr oft, wenn zum Beispiel Eltern das Eigentum an die Kinder übertragen im Gegenzug zur Eintragung eines Nießbrauches zu ihren Gunsten. Dies ist sinnvoll, da sich der Nießbrauch in einem wichtigen Punkt von den bisher genannten Rechten unterscheidet: Der Nießbrauch ist an eine konkrete Person gebunden, also nicht vererbbar. Es ist ein Recht, das an eine aufhebende Bedingung geknüpft ist, in diesem Fall an die Lebenszeit des Berechtigten.

2.7.1.2 Räumliche Veränderungen

Ein Grundstück ist ein Teil der Erdoberfläche, der im Grundbuch unter einer eigenen Nummer verzeichnet ist (Eickmann 1986, S.44). Das Grundbuch verwaltet also nicht das Grundstück selbst, sondern nur einen eindeutigen Bezeichner. Ein Grundstück ist bestimmt durch seine Identität. Diesem Grundstücksbegriff entsprechend werden räumliche Veränderungen durch Veränderungen an den Identitäten der Grundstücke abgebildet.
Räumliche Veränderungen im Grundbuch entstehen durch *Vereinigung* oder *Teilung* von Grundstücken oder durch *Bestandszuschreibung*.
Räumliche Veränderungen an Grundstücken beeinflussen nicht nur die Grundstücke selbst, sondern auch die Rechtsverhältnisse an Grundstücken. Was gilt etwa für eine Hypothek, wenn das Grundstück, das sie belastet, mit einem anderen Grundstück vereinigt wird? Ist die Vereinigung überhaupt möglich? Grundsätzlich kann gesagt werden, daß räumliche Operationen immer dann möglich sind, wenn Rechtsverhältnisse nicht unklar werden, wenn also „hiervon Verwirrung nicht zu besorgen ist“ (§5 GBO). In dem obigen Beispiel ist das dann der Fall, wenn das Objekt der Belastung nach der Vereinigung noch eindeutig zu identifizieren ist (Spezialitätsprinzip). Dies ist gegeben durch die Tatsache, daß die Flurstücke, die das alte Grundstück bildeten, noch vorhanden sind.

- Die Grundstücksvereinigung

Durch die Vereinigung zweier benachbarter Grundstücke entsteht ein neues Grundstück, das die beiden alten Grundstücke umfaßt. Grund hierfür kann sein, daß eine Person ein Teil eines Nachbargrundstücks kauft und mit dem seinen verbindet. Voraussetzung ist, daß beide Grundstücke derselben Person gehören. Belastungen an den ehemaligen Einzelgrundstücken bleiben an diesen bestehen, das heißt an den Flurstücken, aus denen das Einzelgrundstück bestand.

- Die Grundstücksteilung

Ein Grundstück kann geteilt werden. Daraufhin entstehen zwei neue Grundstücke, die zusammen die Fläche des alten umfassen. Es könnte zum Beispiel sein, daß eine Person ein Teil ihres Grundstück an einen Nachbarn verkaufen will. Dazu muß das Grundstück geteilt werden in den Teil, den die Person behält, und den Teil, den sie zu verkaufen beabsichtigt.

- Die Bestandszuschreibung

Ein Grundstück kann per Bestandszuschreibung einem anderen Grundstück zugeordnet werden. Dabei wird das Grundstück Teil des anderen und hört selbst auf zu bestehen. Das Grundstück, dem es zugeschrieben wurde, umfaßt dann die Fläche beider Grundstücke. Dies

könnte zum Beispiel dann der Fall sein, wenn, entsprechend dem obigen Beispiel, eine Person einen Teil des Grundstücks vom Nachbarn erwirbt und diesen Teil dann per Bestandszuschreibung dem eigenen Grundstück hinzufügt. Für Belastungen an den vormaligen Einzelgrundstücken gilt: Belastungen am zugeschriebenen Grundstück bleiben an ihm bestehen, während sich Belastungen des Hauptgrundstücks nach der Zuschreibung auch auf das zugeschriebene Grundstück erstrecken.

2.7.1.3 Beziehungen zwischen Operationen in Grundbuch und Kataster

Entsprechend den Beziehungen zwischen Kataster und Grundbuch können die räumlichen Operationen im Grundbuch nicht losgelöst vom zugehörenden Kataster betrachtet werden. Zu den räumlichen Operationen im Grundbuch bestehen duale Operationen im Kataster. Die der Grundstücksvereinigung entsprechende Operation ist die Verschmelzung zweier Flurstücke. Der Grundstücksteilung entspricht die Zerlegung. Nun müssen sich diese Begriffe nicht zwangsläufig decken. Eine Grundstücksvereinigung verursacht nicht notwendigerweise eine Verschmelzung der Flurstücke. Lediglich für den Fall, daß ein Idealgrundstück gebildet werden soll, müssen die Flurstücke verschmolzen werden. Soll ein Grundstück geteilt werden, dann ist eine Zerlegung nur dann notwendig, wenn der vom Grundstück zu teilende Teil nicht einem Flurstück entspricht, aus dem das Grundstück besteht.

2.7.2 Abläufe

2.7.2.1 Der Charakter von Abläufen im Grundbuchsystem

Veränderungen im Grundbuchsystem sind die Folge von komplexen Abläufen. Die bisher beschriebenen Operationen sind elementar und bilden die Bestandteile, aus denen sich die Abläufe zusammensetzen. Abläufe sind aber nicht nur gekennzeichnet durch ihre Komplexität, sondern auch durch ihre zeitliche Ausdehnung. Das Grundbuchverfahren erfordert die Zusammenarbeit verschiedener Menschen und Behörden. Die Bearbeitung durch die Angestellten des Grundbuchamts nimmt gewisse Zeit in Anspruch. Den Zustand des Grundbuchsystems kennzeichnen solche Abläufe, die abgeschlossen oder nicht abgeschlossen sein können. Abläufe unterliegen äußeren Einflüssen. Das Grundbuchverfahren wird zum Beispiel beeinflußt von Gerichtsentscheidungen, anderen menschlichen Entscheidungen, etwa der Bewertung eines Beweismittels, sowie von der Arbeit anderer Behörden, wie des Katasteramts. Diese äußeren Einflüsse werden von einem Ablauf aufgenommen und beeinflussen seinen Verlauf.

2.7.2.2 Klassifikation von Abläufen

Komplexe Abläufe im Grundbuchsystem sind Grundbuchverfahren. Sie repräsentieren nicht nur die Operationen auf den Grundbuchdaten, sondern auch die Arbeitsweise des Grundbuchamts, seine Entscheidungen, die Aufnahme äußerer Einflüsse sowie die zeitliche Ausdehnung des Verfahrens. Zu der Spezifizierung eines Grundbuchverfahrens gehört folglich die Beschreibung der darin vorkommenden Operationen in Abhängigkeit der äußeren Einflüsse. Ziel eines Grundbuchverfahrens ist es, eine bestimmte Eintragung im Grundbuch auszulösen. Dies wird erreicht durch die Herbeiführung der Operation, die zu der Eintragung führt. Komplexe Abläufe können daher entsprechend der Operation, die als Ziel angestrebt wird, eingeteilt werden. Um dieses Ziel zu erreichen, ist es notwendig, die Vorbedingungen für diese Operationen zu erfüllen. Alle anderen Operationen in diesem Ablauf dienen der Erfüllung dieser Vorbedingungen. Entsprechend dieser Argumentation kann jeder elementaren Operation ein komplexer Ablauf zugeordnet werden, der die Arbeit des Grundbuchamts repräsentiert.

2.7.3 Äußere Einflüsse

Zur Beschreibung der Abläufe ist es unumgänglich, auf sie wirkende äußere Einflüsse genauer zu betrachten. Ausgangspunkt ist die Tatsache, daß Grundbuchverfahren von Abläufen beeinflußt werden, die nicht der Kontrolle des Systems unterliegen. Folglich können sie auch nicht vollständig spezifiziert werden. Diese Abläufe sind die menschlichen Entscheidungen. Nun sind diese Einflüsse nicht beliebig. Zwar sind sie selbst nicht vollständig spezifizierbar und es ist nicht vorhersehbar, wann sie auftreten, aber ihre möglichen Auswirkungen auf das Grundbuch sind Regeln unterworfen. Diese Wirkungen lassen sich als Folge elementarer Operationen beschreiben, mit dem Unterschied, daß sie nicht durch Abläufe innerhalb des Systems verursacht werden, sondern durch äußere Abläufe. Im folgenden werden diese Art elementarer Operationen als *externe Operationen* bezeichnet im Gegensatz zu *internen Operationen*. Ein weiterer Unterschied der externen Operationen liegt darin, daß sie nicht Änderungen in den Grundbuchdaten selbst verursachen, sondern lediglich die Vorbedingungen für die Durchführung solcher Eintragungen beeinflussen. Beispielsweise wird als Folge der Beweiswürdigung extern die Registrierung eines Kaufvertrags verursacht. Dies selbst erzeugt noch keine Eintragung eines Eigentumstransfers im Grundbuch, schafft aber eine wesentliche Voraussetzung hierfür. Die im vorangegangenen Abschnitt gegebene Charakterisierung von Abläufen kann also in der folgenden Weise ergänzt werden: Äußere Einflüsse werden im Grundbuchsystem durch elementare Operationen abgebildet, die nicht vom System selbst ausgelöst werden und direkt keine Veränderungen im Grundbuch bewirken.

2.8 Inkonsistenzen

Bei der Repräsentation eines Grundbuchsystems, gerade unter der Bedingung ständiger Veränderungen, erfordert die Erhaltung der Konsistenz des Systems erhöhte Anstrengungen. Nun hat der Gesetzgeber das Grundbuch zu Gunsten der Schaffung von Rechtssicherheit mit dem öffentlichen Glauben ausgestattet. Damit gilt der Inhalt des Grundbuchs nach außen als richtig. Dies bedeutet aber nicht, daß aus der möglichen Nichtübereinstimmung mit dem materiellen Recht keine großen Probleme entstehen können. Einerseits wird Gerechtigkeit zu Gunsten der Rechtssicherheit hinten angestellt und andererseits entstehen beträchtliche Schadenersatzforderungen gegenüber dem Staat. Es ist deshalb sehr wichtig, Inkonsistenzen zu vermeiden. Um dies zu erreichen, ist vorher eine genaue Kenntnis der Arten von Inkonsistenzen und der Quellen ihrer Entstehung notwendig.

2.8.1 Inkonsistenz und Grundbuchunrichtigkeit

Bevor Inkonsistenzen betrachtet werden können, muß entschieden werden, welches Konzept von Konsistenz in dem betrachteten System anwendbar ist. Was bedeutet also Konsistenz in Bezug auf ein Grundbuchsystem? Zur Untersuchung dieser Frage muß von den Aufgaben eines Grundbuchsystems ausgegangen werden: Es dient dazu, über die Rechtsverhältnisse an Grundstücken Auskunft zu geben (Löffler 1993, S.11). Das bedeutet, daß die materiellen Rechtsverhältnisse im Grundbuch abgebildet werden müssen. Weiterhin bedeutet dies, daß sich die Abbildung der Rechtsverhältnisse im Grundbuch mit dem materiellen Recht in Übereinstimmung befinden muß. Dementsprechend kann der Begriff der Konsistenz in Bezug auf ein Grundbuchsystem wie folgt definiert werden: *Das Grundbuchsystem ist konsistent, wenn sich die aus ihm hervorgehenden Rechtsverhältnisse mit dem materiellen Recht decken.* Umgekehrt kann die Inkonsistenz als Nichtübereinstimmung von Grundbuch und materiellem Recht bezeichnet werden. Im Grundbuchrecht hat dieser Gedanke Eingang gefunden in den Begriff der *Grundbuchunrichtigkeit:* Entsprechend §894 BGB ist das Grundbuch unrichtig, wenn sein Inhalt mit der tatsächlichen Rechtslage nicht übereinstimmt. Entsprechend dieser Argumentation decken sich Inkonsistenz und Grundbuchunrichtigkeit: Das Grundbuch ist inkonsistent, wenn es unrichtig ist.

2.8.2 Arten der Grundbuchunrichtigkeit

Abhängig von dem Zeitpunkt ihrer Entstehung wird die Grundbuchunrichtigkeit in zwei Kategorien eingeteilt (Demharter 1997, S.333): die *ursprüngliche* Unrichtigkeit und die *nachträgliche* Unrichtigkeit. Ursprüngliche Unrichtigkeit bedeutet, daß die betreffende Eintragung von vornherein, also zum Zeitpunkt ihrer Eintragung, unrichtig war. Sie kann zum Beispiel eintreten, wenn bestimmte Verfahrensvoraussetzungen nicht erfüllt wurden oder wenn das Objekt des Rechtsgeschäfts verwechselt wurde, etwa durch Angabe einer falschen Flurstücksnummer. Die zweite Kategorie, die nachträgliche Unrichtigkeit, wird auch als *Unvollständigkeit* bezeichnet. Sie tritt ein, wenn das Grundbuch von vornherein richtig war, aber dann durch Rechtsvorgänge außerhalb nachträglich unrichtig wird. Beispiel hierfür ist etwa der Übergang von Eigentum bei Erbfolge. Im folgenden werden die Abläufe, die zu der Entstehung von Grundbuchunrichtigkeit führen, entsprechend dieser Einteilung näher beschrieben.

2.8.3 Ursprüngliche Grundbuchunrichtigkeit

In diesem Abschnitt wird beschrieben, welche Abläufe die ursprüngliche Grundbuchunrichtigkeit verursachen.

- Verletzung gesetzlicher Vorschriften durch das Grundbuchamt

Zwar unterliegt die Arbeit des Grundbuchamts den gesetzlichen Vorschriften. Auch ist es entsprechend dem Legalitätsprinzip verpflichtet, die Richtigkeit des Grundbuchs zu erhalten. Dennoch passieren bei der menschlichen Arbeit Fehler, die zur Verletzung der gesetzlichen Vorschriften führen. Als Beispiele sollen genannt werden: Die Teilung eines Grundstücks ohne den Antrag des Eigentümers und das mögliche Nichtvorliegen der Antragsvoraussetzungen.

- Unrichtigkeit in Folge des formellen Konsensprinzips

Das formelle Konsensprinzip besagt, daß zur Herbeiführung einer Eintragung im Grundbuch nur die einseitige Erklärung dessen notwendig ist, dessen Recht von der Eintragung betroffen ist (Bewilligung). Dies entbindet das Grundbuchamt von der Pflicht zu prüfen, ob der Rechtsübergang auch tatsächlich stattgefunden hat. Hieraus entsteht die Möglichkeit, daß Grundbucheintrag und materielles Recht auseinanderlaufen.

- Verwechslungen bezüglich des Rechtsgegenstands.

Hierzu zählt die Verwechslung durch die Beteiligten an einem Rechtsgeschäft und die Verwechslung vor oder bei der Erstellung des Grundbuchs. In die erste Kategorie fällt die Verwechslung von Flurstücksnummern bei einem Rechtsgeschäft. Zwar erfolgt eine Eintragung in das Grundbuch über einen Rechtsübergang, jedoch fehlte die materiellrechtliche Voraussetzung für den Rechtsübergang, da sich die Parteien über das Objekt des Rechtsverkehrs geirrt haben. Zur zweiten Kategorie gehören die Fälle, in denen bei der Erstellung des Grundbuchs oder im Vorfeld zum Beispiel die Flurstücke zweier benachbarter Grundstücke verwechselt wurden.

2.8.4 Nachträgliche Grundbuchunrichtigkeit

Inhalt dieses Abschnittes ist die Beschreibung der Abläufe, die zu einer nachträglichen Grundbuchunrichtigkeit führen.

- Übergang von Rechten außerhalb des Grundbuchs

Rechte können auch außerhalb des Grundbuchs übergehen, zum Beispiel durch Erbgang.

- Änderung der gesetzlichen Grundlagen

Das Grundbuch ist konsistent, wenn es mit der materiellen Rechtslage übereinstimmt. Nun ist es aber denkbar, daß sich die Rechtslage ändert, indem neue Gesetze in Kraft treten. Dann kann das Grundbuch bezüglich dieser neuen Rechtslage inkonsistent sein, obwohl es vorher konsistent war. Ein Beispiel für solche Gesetzesänderungen in der jüngeren Geschichte war der Übergang des Eigentums der ehemaligen DDR an die Bundesrepublik Deutschland infolge des Einigungsvertrags.

- Gegenstandslosigkeit von Angaben

Das Grundbuch kann dadurch unrichtig werden, daß in ihm eingetragene Rechte gegenstandslos werden (Bengel und Simmerding 1995, S.419). Dies ist dann der Fall, wenn sie nicht mehr auszuüben sind und sich das auch in Zukunft nicht ändern wird. Das kann dadurch passieren, daß der Gegenstand des Rechts in der Örtlichkeit nicht mehr auffindbar oder der Berechtigte, also der Inhaber des Rechts, nicht mehr zu ermitteln ist. Denkbar ist im ersten Fall, daß ein Grundstück überflutet worden ist.

- Das Erlöschen eines an eine Bedingung geknüpften Rechts.

Ein Recht kann zum Beispiel auf die Lebenszeit des Berechtigten begrenzt sein. Mit dem Tod dieser Person wird das Grundbuch unrichtig, da die Bedingung, an die das Recht geknüpft war, nicht mehr erfüllt ist. Beispiel hierfür ist der Nießbrauch.

- Inkonsistente Rangverhältnisse

Entsprechend dem Prioritätsprinzip bestimmen die Reihenfolge der Eintragungen oder das Datum ihrer Eintragung die Rangverhältnisse der Rechte. Durch verschiedene Eintragungen, Löschungen und eventuelle nachträgliche Rangveränderungen können die Rangverhältnisse der Rechte untereinander unklar werden. Unklarheit besteht dann, wenn die Eintragungen zu Zweifeln an den materiellen Rangverhältnissen führen (Bengel und Simmerding 1995, S.423). Denkbar sind vor allem Fälle, in denen der Rang einer Eintragung relativ zu einer anderen Eintragung festgelegt wurde. Existieren mehrere solche relativen Rangverhältnisse, kann es Zweifel über das Rangverhältnis dieser Rechte zueinander geben.

2.8.5 Die Ursachen der Grundbuchunrichtigkeit

Die Grundbuchunrichtigkeit hat zwei Ursachen, die im folgenden erläutert werden.

- Unvollständigkeit der Repräsentation

Das Grundbuch ist eine Abbildung oder Repräsentation der materiellen Rechtsverhältnisse an Grundstücken. Eine solche Repräsentation hat Schwachstellen, da sich nicht alle Aspekte und Einflüsse, welche die materiellen Rechtsverhältnisse bestimmen, vollständig abbilden lassen. Dies hat mehrere Gründe. Erstens ist das System historisch gewachsen und somit auch von historisch gewachsenen Arbeitsweisen und Traditionen geprägt. Zweitens ist es, um die Arbeitsfähigkeit des Grundbuchamts sicherzustellen, notwendig, sich auf die wesentlichen Aspekte zu beschränken, da sonst das System zu komplex wird. Drittens lassen sich in einer Welt, die sich ständig verändert und die sehr komplex ist, nicht alle relevanten Aspekte abbilden. Dies liegt auch daran, daß sich diese Einflüsse ständig verändern. Die Repräsentation der materiellen Rechtsverhältnisse und der ihre Veränderung bewirkenden Einflüsse muß also unvollständig bleiben. Diese Unvollständigkeit der Repräsentation ist eine Ursache für die Grundbuchunrichtigkeit. Insbesondere die Tatsache, daß sich die Rechtsverhältnisse außerhalb der Repräsentation ändern, bewirkt Unrichtigkeiten im Grundbuch. Die im vorigen Abschnitt geschilderten nachträglichen Unrichtigkeiten sind durch dieses Problem verursacht. Ein Beispiel für den Kompromiß zwischen Arbeitsfähigkeit des Grundbuchamts und Abbildung aller Aspekte des materiellen Rechts ist das formelle Konsensprinzip. Die Probleme, die in diesem Zusammenhang bestehen, werden dadurch offenbar, daß infolge dieses Kompromisses das Grundbuch unrichtig werden kann (in Form der ursprünglichen Unrichtigkeit).

- Fehlfunktion des Systems

Die zweite Ursache für die Grundbuchunrichtigkeit ist die Tatsache, daß trotz aller gesetzlichen Regelungen Fehler unterlaufen. Wo Menschen handeln, sind Fehler nicht zu vermeiden, sowohl auf der Seite des Grundbuchamts, das dadurch möglicherweise Gesetze verletzt, als auch auf der Seite der Personen, die Rechtsgeschäfte führen. Dies kann durch Unachtsamkeit, Unwissenheit oder aus anderen Gründen geschehen, vollständig ausschließen läßt es sich nicht. Folglich entstehen durch diese Ursache Unrichtigkeiten im Grundbuch, vorrangig in der Form der ursprünglichen Grundbuchunrichtigkeit, da diese Fehler meist unmittelbar wirken.

2.8.6 Die Korrektur des unrichtigen Grundbuchs

Nach der Beschreibung der Arten der Grundbuchunrichtigkeit und der Ursachen ihrer Entstehung werden nun die im Grundbuchrecht vorgesehenen Möglichkeiten zu ihrer Korrektur beschrieben.
Prinzipiell bestehen fünf Wege, das Grundbuch zu berichtigen. Es ist möglich, das Grundbuch auf Antrag eines Beteiligten zu berichtigen. Dies gibt dem Eigentümer die nötigen Mittel in die Hand, sich gegen die Möglichkeit des gutgläubigen Erwerbs zu wehren. Die zweite Möglichkeit ist die Berichtigung von Amts wegen, das heißt ohne einen Antrag, durch das Grundbuchamt selbst. Die dritte Möglichkeit liegt zwischen diesen beiden. So kann das Grundbuchamt in bestimmten Fällen die Beteiligten zu einem Antrag auf Grundbuchberichtigung verpflichten. Weiterhin kann das Grundbuch von gegenstandslosen Eintragungen befreit werden, indem diese gelöscht werden. Eine fünfte Möglichkeit, die zur Richtigkeit des Grundbuchs führt, ist der Erwerb durch einen gutgläubigen Dritten. Dies ist natürlich nicht die anzustrebende Lösung, drohen doch Schadenersatzforderungen des wirklich Berechtigten, dessen Recht aus dem Grundbuch nicht korrekt ersichtlich war. Deshalb weicht der Gesetzgeber in den Fällen, in denen gutgläubiger Erwerb droht, vom Antragsprinzip ab und ermächtigt das Grundbuchamt, ohne Antrag tätig zu werden.

- Gutgläubiger Erwerb

Geht aus dem Grundbuch die falsche Rechtslage hervor, so kann die Gefahr des gutgläubigen Erwerbs bestehen. Findet gutgläubiger Erwerb statt, beseitigt dies die Grundbuchunrichtigkeit, die eventuell bestanden hat. Der Erwerber wird rechtmäßiger Eigentümer. Dies sieht der Gesetzgeber zur Schaffung von Rechtssicherheit so vor. Da aber dadurch Schadenersatzansprüche gegenüber dem Staat entstehen und der vorher wirklich Berechtigte sein Recht verliert, stellt diese Lösung die Ausnahme dar.

- Berichtigung auf Antrag

Nach §894 des BGB kann derjenige, dessen Recht nicht oder nicht richtig eingetragen ist oder dessen Recht durch eine falsche Eintragung beeinträchtigt ist, die Zustimmung zur Grundbuchberichtigung von demjenigen verlangen, dessen Recht durch die Eintragung der Berichtigung betroffen ist (Bengel und Simmerding 1995, S.292). Dies öffnet einen Weg zur Berichtigung des Grundbuchs, die Berichtigung durch Vorlage einer Berichtigungsbewilligung. Das heißt, daß ein Antrag auf Berichtigung gestellt und die Unrichtigkeit dadurch glaubhaft gemacht wird, daß das Einverständnis zur Berichtigung durch die Person vorliegt, deren Recht die Berichtigung eingeschränkt. Die zweite Möglichkeit der Berichtigung wird begründet durch §22 GBO, nachdem die Bewilligung nicht notwendig ist, wenn die Unrichtigkeit nachgewiesen werden kann.

- Der Berichtigungszwang

In den Fällen, in denen das Grundbuch durch Rechtsvorgänge außerhalb unrichtig wird, besteht die Möglichkeit, daß das Grundbuchamt die Berechtigten verpflichtet, einen Antrag auf Berichtigung des Grundbuchs zu stellen. Diese Möglichkeit basiert auf §82 GBO. Wenn eine solche Verpflichtung nicht möglich, also zum Beispiel kein Beteiligter aufzufinden ist, so kann die Berichtigung auch in einem Amtsverfahren, also ohne Antrag erfolgen (§82a GBO).

- Berichtigung von Amts wegen

Hat das Grundbuchamt gesetzliche Vorschriften verletzt, so besteht die Möglichkeit, daß es auch ohne Antrag auf dem Wege eines Amtsverfahren tätig wird. Dies ist in §53 GBO geregelt. Danach hat das Grundbuchamt einen *Amtswiderspruch* einzutragen, wenn eine Grundbuchunrichtigkeit besteht und das Grundbuchamt gesetzliche Vorschriften verletzt hat und die Möglichkeit gutgläubigen Erwerbs besteht. Damit ist die Unrichtigkeit gekennzeichnet, und es besteht nicht die Möglichkeit des gutgläubigen Erwerbs und folglich auch nicht die Gefahr von Schadenersatzansprüchen gegenüber dem Staat (Eickmann 1986, S.262). Weiter heißt es in §53 GBO, daß eine Eintragung von Amts wegen zu löschen ist, wenn sie inhaltlich unzulässig ist.

- Die Löschung gegenstandsloser Eintragungen

Wenn Eintragungen gegenstandslos werden, kann das Grundbuchamt sie löschen (§84 GBO). Prinzipiell entscheidet das Grundbuchamt nach freiem Ermessen, wann eine gegenstandslose Eintragung zu löschen ist. Dies soll jedoch nur geschehen, wenn ein hinreichender Anlaß gegeben ist (§85 GBO). Solche Anlässe sind zum Beispiel die Umschreibung eines Grundbuchblatts infolge Unübersichtlichkeit oder die Anregung durch einen Beteiligten.

2.9 Zusammenfassung

In diesem Kapitel sind Aufbau und Arbeitsweise des Grundbuchamts beschrieben worden. Dabei wurde ausgegangen von Grundprinzipien, die im Grundbuchrecht allgemein anerkannt sind. Die Arbeitsweise des Grundbuchamts kann zufriedenstellend beschrieben werden, wenn zwei Ebenen dabei betrachtet werden: Die Ebene der elementaren Operationen und die der komplexen Abläufe. Elementare Operationen beschreiben die eigentlichen Eintragungen in das Grundbuch mit ihren Voraussetzungen und Wirkungen. Komplexe Abläufe beschreiben die Arbeitsweise des Grundbuchamts zur Prüfung bzw. Schaffung der Voraussetzungen für die Eintragungen in das Grundbuch. Komplexe Abläufe treten im Grundbuchrecht als Grundbuchverfahren in Erscheinung. Von großer Bedeutung für die Repräsentation eines Grundbuchsystems ist die Kenntnis der kritischen Stellen, die Fehler verursachen. Der Begriff der Inkonsistenz im Grundbuch wird mit dem Begriff der Grundbuchunrichtigkeit identifiziert. Deshalb wurde der Untersuchung der Grundbuchunrichtigkeit, ihrer Quellen und Ursachen sowie der im Grundbuchrecht verankerten Möglichkeiten zu ihrer Beseitigung Raum gegeben.

3 Das konzeptionelle Modell

3.1 Einleitung

Basierend auf der Analyse des Grundbuchs im vorigen Kapitel wird nun das Modell eines Grundbuchsystems erstellt. Einerseits ist es dabei wichtig, die wesentlichen Aspekte und Probleme abzubilden. Auf der anderen Seite soll das Modell auch möglichst klein sein, damit es übersichtlich bleibt. Deshalb liegt der Schwerpunkt dieses Kapitels nicht in einer vollständigen Modellierung des gesamten Grundbuchs, sondern in der Modellierung der wichtigen und aussagekräftigen Fälle. Ausgegangen wird dabei von der Frage, in welchem Verhältnis das zu erstellende Modell, das Grundbuch und das materielle Recht zueinander stehen. Daraus ergibt sich, was modelliert werden muß, was die wesentlichen Aspekte sind und was folglich Inhalt des Modells sein muß. Um ein Modell zu erstellen, ist es notwendig, die bei der Repräsentation auftretenden Probleme zu analysieren und die Ergebnisse in die Modellierung einfließen zu lassen. Das konzeptionelle Modell beschreibt die Objekte und Relationen, die für das Grundbuchsystem charakteristisch sind. Ein zentraler Punkt dieser Arbeit ist die Darstellung der Veränderungen des Systems in der Zeit. Dazu werden die Operationen und Abläufe beschrieben, die diese Veränderungen verursachen. Abschließend wird das Modell hinsichtlich seines Verhältnisses zum materiellen Recht und zum Grundbuch diskutiert.
Das konzeptionelle Modell in diesem Kapitel spannt den Bogen von der allgemeinen Beschreibung des Grundbuchs aus Kapitel 2 zu dem Ausschnitt des Grundbuchs, der modelliert werden soll. Dieser Ausschnitt des Grundbuchs wird rein informell dargestellt und legt damit den Inhalt des Modells fest. Es schafft also die konzeptionellen Voraussetzungen für die formale Beschreibung.

3.2 Konzeptionelles Modell, Grundbuch und materielles Recht

In welchem Zusammenhang stehen das materielle Recht, das Grundbuch und das konzeptionelle Modell? Das materielle Recht stellt die Rechtsvorgänge dar, die in der Welt ablaufen. Das Grundbuch ist dazu geschaffen, diese Rechtsverhältnisse zu verwalten, es stellt also eine Abbildung dieser Rechtsverhältnisse dar. In welchem Verhältnis steht dazu das Modell, das in dieser Arbeit entwickelt werden soll? Das Modell soll ebenfalls die materiellen Rechtsverhältnisse abbilden. Es ist also in diesem Sinne eine Repräsentation dieser Rechtsverhältnisse, wie das Grundbuch selbst. Wie ist nun das Verhältnis dieser beiden Repräsentationen? Das hier zu erstellende Modell kann nicht losgelöst von der Grundbuchrepräsentation lediglich in Bezug auf die Abbildung der materiellen Rechtsverhältnisse betrachtet werden. Das liegt daran, daß vom Gesetzgeber ein Verfahrensrecht geschaffen wurde, das die konsistente Verwaltung der materiellen Rechtsverhältnisse im Grundbuch ermöglichen soll. Diese rechtlichen Grundlagen gelten auch gegenüber jeder anderen Repräsentation. Das zu erstellende Modell muß also sowohl ein Modell der materiellen Rechtsverhältnisse als auch des Verfahrensrechts sein.
Wie kann dies erreicht werden? Die Analyse des Grundbuchsystems legte offen, daß das Grundbuch nicht in der Lage ist, die Grundbuchunrichtigkeit generell zu verhindern. Folglich ist das Grundbuch keine korrekte Repräsentation der materiellen Rechtsverhältnisse. Dies bedeutet aber auch, daß eine Repräsentation des Grundbuchs mit diesem Problem behaftet ist. Wie läßt sich der Widerspruch überbrücken zwischen einerseits der Modellierung des Grundbuchs und andererseits der (korrekten) Modellierung der materiellen Rechtsverhältnisse? Die Lösung liegt hier in der Kennzeichnung dieses Widerspruchs, das heißt in der Kennzeichnung der inkonsistenten Stellen. Dies stellt insofern eine Lösung dar, als daß weder die materiellen Rechtsverhältnisse noch das Grundbuchrecht verändert werden können und folglich das Aufzeigen der Konfliktstellen das Maximum ist, das erreicht werden kann. Diese Gedanken können umgesetzt werden, indem das hier zu erstellende Modell das

Grundbuch zum einen als Repräsentation der materiellen Rechtsverhältnisse abbildet, zum anderen aber auch die materiellen Rechtsverhältnisse selbst. Dies ermöglicht die Betrachtung des Verhältnisses dieser Rechtsverhältnisse und ihrer Abbildung im Grundbuch.

3.3 Der Inhalt des Modells

Das Modell zielt darauf, die wesentlichen Aspekte der Arbeit eines Grundbuchsystems abzubilden. Was sind aber die wesentlichen Aspekte? Ausgehend von der Analyse des Grundbuches in Kapitel 2 ist es für eine korrekte Abbildung von zentraler Bedeutung, die Veränderungen des Systems in der Zeit zu beschreiben. Dies muß unter Beachtung der dabei möglicherweise auftretenden Inkonsistenzen geschehen, die in Folge der Nichtübereinstimmung von materiellem Recht und Grundbuch entstehen. Der vorige Abschnitt motivierte, daß dies durch Modellierung des Grundbuchs und der materiellen Rechtsverhältnisse geschehen muß. Das Modell wird in drei Schritten erstellt:

1. Die Modellierung der materiellen Rechtsverhältnisse,
2. die Modellierung des Grundbuchs,
3. die Beschreibung der Beziehungen zwischen 1. und 2.

Der erste Schritt der Modellierung beschreibt die materiellen Rechtsänderungen, die ihre Abbildung im Grundbuch finden. Insofern wird die zu modellierende Welt Teil des Modells mit dem Vorteil, Aussagen über das Verhältnis der Welt und ihrer Repräsentation im Grundbuch treffen zu können. Diese Aussagen werden dann im Schritt 3 beschrieben. Sie beziehen sich auf die Korrektheit der aus dem Grundbuch ersichtlichen Rechtslage bezüglich des Modells des materiellen Rechts. Die Stellen, an denen Grundbuch und materielles Recht nicht übereinstimmen, werden markiert.

Zur Modellierung des Grundbuchs im Schritt 2 müssen die Objekte und Operationen im System beschrieben werden. Ausgehend von der Analyse der Veränderungen in einem Grundbuchsystem müssen die Operationen auf zwei Ebenen beschrieben werden, auf der Ebene der elementaren Operationen und der Ebene der komplexen Abläufe. Dabei ist es notwendig, ihre Vorbedingungen und Wirkungen anzugeben. Vorbedingungen sind wichtig, da nicht in jeder Situation jede Operation möglich ist. Um das System konsistent zu beschreiben, müssen die Wirkungen der einzelnen Operationen genau angegeben werden. Welche dies sind, legt das Grundbuchrecht fest.

Dieses Modell beschreibt die Rechtsverhältnisse an Grundstücken und ihre Veränderung. Es beinhaltet nicht den Zusammenhang mit dem Kataster, das Grundlage der Darstellung der Grundstücke im Grundbuch ist. Somit zielt es auch nicht auf die Diskussion der Probleme, die aus der räumlichen Lage, den möglichen Inkonsistenzen zwischen der Lage in der Örtlichkeit und der Darstellung in Kataster und Grundbuch erwachsen. Zu einer ausführlichen Diskussion dieser Probleme wird verwiesen auf (Bengel und Simmerding 1995).

3.4 Das Modell der materiellen Rechtsverhältnisse

Am Anfang des konzeptionellen Modells steht die Beschreibung der materiellen Rechtsverhältnisse, also der Welt, die im Grundbuch abgebildet werden soll. Dies hat zwei Gründe: Erstens legt die Beschreibung der materiellen Rechtsverhältnisse die Grundlage für ihre Abbildung im Grundbuch. Das heißt, daß sie die Fälle festlegt, die in diesem Modell behandelt werden. Dies entspricht dem Gedanken, im Modell nicht das komplette Grundbuch abzubilden, sondern nur die interessanten Fälle, die das Grundbuchsystem charakterisieren. Das bedeutet auch, daß nur der Teil der Welt, der im Modell des Grundbuchs abgebildet wird, auch Teil der Modellierung der Welt ist. Zweitens ist die Beschreibung der materiellen Rechtsverhältnisse Grundlage dafür, Aussagen über das Verhältnis von materiellem Recht und seiner Abbildung im Grundbuch zu treffen (siehe Abschnitt 3.2). Insbesondere geht es hierbei um die Aufdeckung von Unrichtigkeiten im Grundbuch. Veränderungen der materiellen Rechtsverhältnisse bilden das Ergebnis von *Aktionen* ab. Zum Beispiel verändert sich der Eigentümer eines Grundstücks in Folge des

Verkaufs des Grundstücks. Der Begriff *Aktionen* wird unterschieden vom Begriff der *Operationen*. Der erste Begriff bezeichnet Veränderungen in der materiellen Welt, der zweite Veränderungen im Grundbuchsystem. Aktionen werden in diesem Sinn als Beobachtungen über die Veränderungen in der materiellen Welt aufgefaßt, als Tatsachen, die in Beziehung zur Abbildung im Grundbuch gesetzt werden.
Diese Beobachtungen müssen mit der Abbildung dieser Veränderungen im Grundbuch in Übereinstimmung sein. Andernfalls besteht eine Grundbuchunrichtigkeit.
Materielle Rechtsverhältnisse werden in diesem Modell als Relationen zwischen Grundstücken, Personen und Rechten dargestellt.
Es stellt sich die Frage, welche Rechte die materiellen Rechtsverhältnisse bestimmen. In Abschnitt 2.7.1.1 wurden Rechte in Voll- und in Teilrechte unterschieden. Teilrechte konnten den Gebrauchswert eines Grundstücks einschränken oder seinen Wiederverkaufswert. Rechte konnten an eine aufhebende Bedingung geknüpft sein. Entsprechend dieser Einteilung werden folgende Rechte modelliert:

1. das Eigentumsrecht als ein Vollrecht,
2. die Hypothek als ein Teilrecht, das den Wiederverkaufswert einschränkt,
3. der Nießbrauch als ein Recht, das den Nutzungswert eines Grundstücks einschränkt und an die Lebenszeit des Eigentümers dieses Rechts geknüpft ist.

3.4.1 Objekte

Bei den Objekten des materiellen Grundstücksverkehrs handelt es sich um Grundstücke und Personen, die Eigentümer von Rechten an diesen Grundstücken sind. Flurstücke bilden die Grundlage der Darstellung der Grundstücke. Materielle Rechtsverhältnisse bestehen folglich zwischen Personen, Grundstücken und Flurstücken.

- *Grundstück*

Ein Grundbuchsystem verwaltet *Grundstücke*. Grundstücke sind die Grundlage des Grundstücksverkehrs.

- *Person*

Personen werden im juristischen Sinn aufgefaßt, das heißt nicht nur Menschen, sondern auch Firmen usw. können Personen sein.

- *Flurstück*

Flurstücke bilden die Grundlage für die Darstellung der Grundstücke. Flurstücke repräsentieren Teile der Erdoberfläche, die im Kataster verwaltet werden.

3.4.2 Relationen

Relationen bezeichnen Beziehungen zwischen den Objekten des materiellen Rechts. Diese Beziehungen betreffen einerseits die materiellen Rechtsverhältnisse und andererseits die Grundlage der materiellen Rechtsverhältnisse, die Grundstücke, das heißt die Beziehungen zwischen Grundstücken und den Flurstücken, aus denen sie gebildet sind.

- *Eigentümer*

Eigentümer ist eine Relation zwischen einer Person und einem Grundstück. Sie bezeichnet die Tatsache, daß der Person das Grundstück gehört.

- *Gläubiger_Hypothek*

Gläubiger_Hypothek ist eine Relation zwischen einer Person, einem Grundstück und einem Wert. Sie bezeichnet die Tatsache, daß die Person Eigentümer einer Hypothek an dem Grundstück ist.

- *Nießbraucher*

Nießbraucher ist eine Relation zwischen einer Person und einem Grundstück. Sie bezeichnet das Recht, ein Grundstück lebenslang zu benutzen.

- *Grundstück_Flurstücke_m*

Grundstück_Flurstücke_m ist eine Relation zwischen einem Grundstück und einem Flurstück. Es bezeichnet die Tatsache, daß das Flurstück Bestandteil des Grundstücks ist.

3.4.3 Aktionen

Aktionen beschreiben die Veränderungen in der materiellen Welt. Sie betreffen die Objekte, also Personen, Grundstücke und Flurstücke.

- *verkaufe_Grundstück*

Eine Person verkauft ein Grundstück an eine weitere Person. Daraufhin wird die weitere Person Eigentümer dieses Grundstücks.

- *bestelle_Hypothek*

Eine Hypothek wird von einer Person zu Lasten des Eigentümers eines Grundstücks bestellt. Daraufhin ist diese Person der Gläubiger dieser Hypothek.

- *zurückzahlen_Hypothek*

Der Eigentümer eines Grundstücks zahlt die Hypothek zurück, die folglich nicht mehr besteht.

- *bestelle_Nießbrauch*

Eine Person, die Eigentümer eines Grundstücks ist, bestellt zu Gunsten einer anderen Person an einem Grundstück einen Nießbrauch.

- *teile_Grundstück*

Der Eigentümer eines Grundstücks teilt sein Grundstück.

- *vereinige_Grundstück*

Der Eigentümer zweier Grundstücke vereinigt sie zu einem Grundstück.

- *stirbt_Person*

Hierunter wird das Ende des Bestehens einer juristischen Person, also sowohl der Tod eines Menschen, als auch die Liquidation einer Firma verstanden. Der Tod einer Person läßt alle an ihre Lebenszeit geknüpften Rechte erlöschen.

3.5 Probleme bei der Repräsentation des Grundbuchs

Um das Grundbuch zu repräsentieren ist es notwendig, die dabei auftretenden Probleme zu analysieren. Im Kapitel 2 wurden die Probleme diskutiert, die im Grundbuch auftreten. Dabei sind einige Probleme der konkreten Art der Repräsentation im Grundbuch geschuldet und andere sind prinzipieller Natur. Die in diesem Modell zu entwickelnde Repräsentation hat ihre spezifischen Probleme, die in diesem Abschnitt dargestellt werden. Dazu gehört die Frage, was Objekte und ihre Identität ausmacht und wie sie sich verändern. Wichtig ist weiterhin zu klären, welche Probleme aus der Interaktion verschiedener Abläufe erwachsen. Im Abschnitt 2.8 ist deutlich geworden, daß sich sehr große Probleme aus der Unvollständigkeit der Repräsentation ergeben. Deshalb werden auch die daraus erwachsenen Probleme für das Modell diskutiert. Eine Hauptquelle für die Grundbuchunrichtigkeit waren Fehlfunktionen des Grundbuchsystems. Was hat dies für Auswirkungen auf das konzeptionelle Modell?

3.5.1 Objekte, Bezeichner und Identitäten

Für die Modellierung eines konsistenten Grundbuchsystems ist es von großer Relevanz zu klären, was die Objekte des Systems sind und was ihre Identität ausmacht. Ist ein Grundstück noch dasselbe Grundstück, wenn es sich räumlich verändert? In Abschnitt 2.6.1 wurde bereits beschrieben, daß ein Grundstück im Grundbuch dadurch repräsentiert ist, daß ihm ein Bezeichner zugewiesen wird. Das Objekt ist also nicht das Grundstück selbst mit seinen Eigenschaften, sondern sein Bezeichner. Vorsicht ist geboten bei der Gleichsetzung von Identität und Bezeichner. Wenn ein Grundstück über einen Bezeichner identifiziert wird, dann ist dabei immer der Zeitpunkt implizit, zu dem dies geschieht. Dies bedeutet, daß der Bezeichner eines Objekts kontextabhängig, also abhängig vom Zeitpunkt ist. Zu unterschiedlichen Zeitpunkten kann ein Bezeichner unterschiedlichen Objekten zugeordnet sein. Ein Objekt hört auf zu bestehen, wenn es keinen Bezeichner hat. Die Identität eines Objekts kann also gleichgesetzt werden mit seinem Bezeichner in einem zeitlichen Kontext. Am Anfang dieses Abschnitts stand die Frage, ob ein Grundstück nach einer räumlichen

Veränderung seine Identität verändert oder nicht. Sie läßt sich nun folgendermaßen beantworten: Es bleibt dasselbe Grundstück, wenn es nach der räumlichen Veränderung noch einen Bezeichner hat. Ob es noch einen Bezeichner hat, ist im Grundbuchrecht festgelegt. Nach einer Vereinigung zum Beispiel verliert das eine Grundstück seinen Bezeichner, während das andere seinen Bezeichner und somit seine Identität behält. Was aber gilt in dem Fall, daß eine Person ihren Namen im Laufe des Lebens verändert? Ändert sich dadurch ihr Bezeichner und somit ihre Identität? Dies ist natürlich nicht beabsichtigt und kann mit Nein beantwortet werden. Der Name einer Person ist eine ihrer Eigenschaften. Dadurch verändert sie nicht ihre Identität, sie bleibt dieselbe Person und folglich muß sie denselben Bezeichner behalten. Aus diesem Beispiel folgt, daß die Eigenschaften eines Objekts durch seinem Bezeichner und dem zeitlichen Kontext bestimmt sind.
Problematisch für die Repräsentation ist die Tatsache daß Bezeichner in einem Grundbuchsystem nicht global eindeutig sind und folglich die Unique Names Assumption (Reiter 1984) verletzt ist (Frank 1996).

3.5.2 Inkonsistenzen durch Abläufe

Eine Quelle von Inkonsistenz erwächst aus der möglichen Interaktion von Abläufen. Zu jedem Zeitpunkt kann es Abläufe geben, die nicht abgeschlossen sind. Diese müssen ebenfalls repräsentiert werden, da das System sonst inkonsistent werden kann. Zum Beispiel besteht die Gefahr gutgläubigen Erwerbs, wenn eine Person ein Grundstück an eine zweite Person verkauft, bevor der erste Käufer sein Recht im Grundbuch eintragen lassen konnte. In diesem Fall kommt es zum Konflikt zwischen zwei Grundbuchverfahren. Das Grundbuchrecht versucht solchen Inkonsistenzen durch Eintragung einer Zwischenverfügung zu begegnen. Sie verhindert den gutgläubigen Erwerb. Nach Abschluß des Verfahrens wird sie wieder gelöscht. Nicht abgeschlossene Abläufe müssen durch die Repräsentation zuverlässig gekennzeichnet werden. Dadurch verhindert man, daß sich die Gefahr von Rechtsverlusten ergibt, wie im obigen Beispiel des gutgläubigen Erwerbs. Nach der Beendigung eines Ablaufs geht das System wieder in einen konsistenten Zustand über.

3.5.3 Die Unvollständigkeit der Repräsentation

Zu den Problemen, die jeder Repräsentation innewohnen, gehört es, daß sie den zu modellierendem Bereich nur unvollständig abbildet. Dies liegt an dem offenen Charakter jedes Teilbereichs der Welt und der damit nicht vollständig zu spezifizierenden Einflüsse auf das zu modellierende System. Im Abschnitt 2.8 wurden die Wirkungen dieses Problems auf die Repräsentation des materiellen Rechts im Grundbuch beschrieben. Die Unvollständigkeit verursacht Unrichtigkeiten im Grundbuch. Nun kann man die äußeren Einflüsse nicht vollständig spezifizieren, sie sind aber dennoch nicht beliebig, wie im Abschnitt 2.7.3 beschrieben wurde. Das konzeptionelle Modell muß Wege für die Aufnahme äußerer Einflüsse und ihre Integration in das System finden. Bemerkt sei an dieser Stelle, daß auf diese Weise Unvollständigkeiten behandelt werden können, die nicht das System als Ganzes verändern, wie etwa Gesetzesänderungen.

3.5.4 Fehlfunktion des Systems

Eine Hauptquelle von Grundbuchunrichtigkeiten ist die Fehlfunktion des Systems in der Gestalt, daß die Regeln des Grundbuchrechts nicht eingehalten werden oder Irrtümer und Verwechslungen vorkommen. Dies ist allerdings keine Frage des konzeptionellen Modells, sondern eine Frage der Umsetzung im formalen Modell bzw. in der Implementation. Das konzeptionelle Modell kann die Voraussetzungen dafür schaffen, daß die Abläufe so reguliert werden, daß solche Probleme vermeidbar sind.

3.6 Das Modell des Grundbuchs

Die vorangegangenen Gedanken fließen in das konzeptionelle Modell ein. Dementsprechend wird nun Schritt 2 (siehe Abschnitt 3.3) umgesetzt, der die Erstellung des konzeptionellen Modells des Grundbuchs beinhaltet. Entsprechend dem Gedanken, die wesentlichen Aspekte des Grundbuchs abzubilden, werden Objekte und die zwischen ihnen bestehenden Relationen festgelegt. Grundlagen hierfür sind die Objekte und Relationen, die das Modell der materiellen Rechtsverhältnisse festlegt. Zur Beschreibung der Veränderungen im Grundbuchsystem werden die Operationen und Abläufe spezifiziert, die charakteristisch für das Grundbuch sind. Es unterscheiden sich interne und externe Operationen. Relationen werden danach eingeteilt, ob sie unmittelbar das materielle Recht abbilden oder ob sie Bestandteil des formellen Rechts sind, also der Dokumentierung der Rechtsverhältnisse dienen. Veränderungen im Grundbuchsystem werden abgebildet als Veränderungen in den Eintragungen. Die entsprechenden Operationen im Grundbuch können also nur diese Eintragungen betreffen. Eintragungen können vorgenommen oder gelöscht werden.

3.6.1 Objekte

Entsprechend der Argumentation in Abschnitt 3.5.1 repräsentiert das Grundbuch die Objekte der materiellen Welt durch ihre Bezeichner und nicht durch die Menge ihrer Eigenschaften. Die Objekte des Grundbuchs, bzw. einer Repräsentation des materiellen Rechts, sind folglich die Bezeichner. In dem Modell werden sie also losgelöst von ihren Eigenschaften (Namen von Personen, usw.) dargestellt. Die Objekte werden hier zusammen mit der intendierten Bedeutung in der materiellen Welt beschrieben. Die Zeichenkette "*ID*" für *Identifier* weist darauf hin, daß die Objekte des Grundbuchs nur Bezeichner für Objekte der materiellen Welt sind.
Die folgenden Objekte sind Abbildungen der Objekte der materiellen Welt. Sie entsprechen also in ihrer intendierten Bedeutung den Objekten der materiellen Welt.
Objekte des Grundbuchs sind:

- *Grundstück_ID,*
- *Person_ID,*
- *Flurstück_ID,* sowie
- *Recht_ID.*

Unter diesen Rechten sind im Grundbuch eintragsfähige Rechte zu verstehen. In diesem Modell werden als Beispiele für eintragsfähige Rechte das Eigentum, die Hypothek und der Nießbrauch dargestellt. Dabei ist Eigentum ein Beispiel für ein Vollrecht und Hypothek ein Beispiel für ein Teilrecht. Beispiel für ein an eine Bedingung geknüpftes Recht ist der auf die Lebenszeit des Berechtigten beschränkte Nießbrauch. Die Einführung dieses Objekts soll der Tatsache Rechnung tragen, daß nicht beliebige Rechte in das Grundbuch eingetragen werden können, sondern nur einige wenige, eben die eintragsfähigen Rechte.

3.6.2 Relationen

Relationen bezeichnen Beziehungen zwischen den Objekten. In diesem konkreten Fall handelt es sich um Beziehungen zwischen Bezeichnern von Objekten in der materiellen Welt. Hierbei werden zwei Arten von Relationen unterschieden: Relationen, die das materielle Recht abbilden, und Relationen, die das formelle Recht abbilden. Das materielle Recht bildet die Rechte an Grundstücken ab. Entsprechend dem Spezialitätsprinzip kann diese Abbildung nicht geschehen ohne die Objekte dieser Rechte, die Grundstücke, genau zu bezeichnen. Deshalb sind die Flurstücke, aus denen ein Grundstück besteht, ebenfalls Teil der Abbildung des materiellen Rechts. Neben der materiellen Rechtslage muß das Modell auch das formelle Recht abbilden, also die Normen, die das Verfahren zur Durchsetzung und Klarstellung der materiellen Rechtslage bestimmen (Bengel und Simmerding 1995, S.132). Hierbei beeinflussen zwei Grundsätze die Arbeit des Grundbuchamts, einerseits der Grundsatz, nur auf Antrag tätig zu werden, und andererseits das Prinzip, daß alle Nachweise in Form beglaubigter

Dokumente erbracht werden müssen. Deshalb verwaltet das Modell die Anträge und Dokumente und bildet ihr Verhältnis zum materiellen Recht ab.

Zuerst werden die Relationen beschrieben, die das materielle Recht abbilden:

- *eingetragenes Recht*

Ein *eingetragenes Recht* ist eine Relation zwischen Bezeichnern von *Grundstücken*, *Personen* und *Rechten*. Ein eingetragenes Recht ist ein Recht, das im Grundbuch zu Gunsten einer Person, an einem Grundstück eingetragen wird.

- *Wert_Recht*

Das *Wert_Recht* ist eine Relation zwischen Bezeichnern von *Rechten*, *Grundstücken*, *Personen* und einem zahlenmäßen *Wert* dieses Rechts. Ein Wert_Recht ist immer genau einem eingetragenen Recht zugeordnet. Ein Wert_Recht bezeichnet den Wert eines eingetragenen Rechtes. Dieser Wert bezieht sich auf ein Recht, das den Wiederverkaufswert beeinträchtigt, da in diesem Fall der Wiederverkaufswert mit der Belastung in Relation gesetzt werden muß. Der Wiederverkaufswert ergibt sich aus dem Wert des Grundstückes abzüglich dieser Belastungen.

- *Grundstück_Flurstücke*

Grundstück_Flurstücke ist eine Relation zwischen Bezeichnern von *Grundstücken* und *Flurstücken*. Diese Relation bezeichnet die Zuordnung der Flurstücke zu Grundstücken, die aus dem Grundbuch ersichtlich ist.

Nun erfolgt die Beschreibung der Relationen, die das formelle Grundbuchrecht abbilden:

- *Dokument*

Ein Dokument ist eine Relation zwischen den Bezeichnern von einem *Grundstück*, einem *Recht* und zwei *Personen*. Es dokumentiert den Übergang des Rechts zwischen diesen Personen und stellt den Beweis für den Übergang dieses Rechts in Form einer Urkunde dar. Im Falle des Eigentumsübergangs repräsentiert dieses Dokument beispielsweise einen Kaufvertrag.

- *Antrag*

Ein *Antrag* ist eine Relation zwischen Bezeichnern eines *Grundstücks*, eines *Flurstücks*, einem *Recht* und zwei *Personen*, von denen eine der Antragsteller ist. Er repräsentiert einen beim Grundbuchamt eingegangenen Antrag. Ein Antrag wird gestellt, um den Übergang eines Rechts zwischen zwei Personen, das ein Grundstück oder auch nur ein Flurstück betrifft, im Grundbuch eintragen zu lassen.

3.6.3 Elementare Operationen

Das Grundbuchsystem bildet Veränderungen im Grundbuch letztlich als Folge von Operationen auf den Daten ab. Die Daten in einem Grundbuchsystem können in zwei Kategorien eingeteilt werden: Einerseits sind es die Daten zur Abbildung des materiellen und andererseits die Daten zur Dokumentierung des formellen Rechts. Dementsprechend können die Operationen danach eingeteilt werden, ob sie das materielle Recht oder das formelle Recht betreffen. Anders gesagt: Sie betreffen Veränderungen in den materiellen Rechtsverhältnissen, oder sie betreffen Beweise zur Dokumentierung dieser Änderungen. In Abschnitt 2.7.3 wurde beschrieben, daß äußere Einflüsse nicht direkt die Abbildung der materiellen Rechtsverhältnisse betreffen. Die elementaren Operationen wurden eingeteilt in interne und externe Operationen. Diese Einteilung kann nun weitergeführt werden. Externe Operationen betreffen das formelle Recht, während interne Operationen die Abbildung der Veränderungen der materiellen Rechtsverhältnisse sind. Interne Operationen wirken folglich auf die Relationen, die das materielle Recht betreffen, externe Operationen dagegen auf die Relationen, die das formelle Recht betreffen.
Die Operationen, die Veränderungen in der Abbildung des formellen Rechts verursachen, bilden die Schnittstelle des Systems nach außen. Sie können mit den externen Operationen

identifiziert werden, während die internen Operationen die sind, die das materielle Recht betreffen.
Im folgenden werden interne und externe Operationen genauer beschrieben, die Inhalt des zu erstellenden Modells sind.
Die möglichen internen Operationen legt das Grundbuchrecht fest. Sie wurden in 2.7.1 beschrieben. Sie entsprechen den Rechtsübergängen, die im Modell der materiellen Rechtsverhältnisse festgelegt wurden. Sie können in rechtliche und räumliche Operationen unterschieden werden. Rechtliche Operationen sind die Eintragung oder Löschung eines Rechts zu Gunsten einer Person an einem Grundstück. Diese Eintragung oder Löschung eines Rechts in einer Datenbank sollen im Kontrast zum Übergang eines Rechts in der materiellen Welt betrachtet werden. Im Grundbuch können Vollrechte und Teilrechte eingetragen oder gelöscht werden. Das Vollrecht ist hierbei das Eigentumsrecht und als Beispiel für ein Teilrecht sollen Hypothek und Nießbrauch dienen. Mit der Eintragung eines Vollrechts verbindet sich notwendigerweise, daß die Eintragung des Voreigentümers gelöscht wird. Weiterhin existieren Operationen zur Eintragung und Löschung eines Nießbrauchs.
Zur zweiten Kategorie gehören die Operationen, die räumliche Veränderungen abbilden. In diesem Modell wird sich beschränkt auf die Grundstücksteilung und die Grundstücksvereinigung. Bei der Modellierung dieser Operationen erfordert die Frage, wie sich die Identitäten der Grundstücke verändern, besondere Aufmerksamkeit. Da im Grundbuch nur die Bezeichner der Grundstücke und nicht die Grundstücke selbst verwaltet werden, müssen sich die Veränderungen in diesen Bezeichnern auswirken. In diesem Modell beschränkt sich die Teilung eines Grundstücks darauf, daß ein Flurstück aus seinem Bestand herausgelöst und als eigenes Grundstück geführt wird. Die Vereinigung zweier Grundstücke führt dazu, daß der Bestand des einen Grundstücks dem anderen hinzugefügt wird. Das bedeutet, daß ein Grundstück dabei seine Identität verliert. Allgemeiner bedeuten die räumlichen Operationen in diesem Modell, daß Grundstücke durch Teilung entstehen und durch Vereinigung mit anderen Grundstücken aufhören zu bestehen.
Welche externen Operationen notwendig sind, wird durch die Frage bestimmt, was zum Nachweis einer materiellen Rechtsänderung an Beweisen und Handlungen notwendig ist. Ausgangspunkt für die Erreichung einer Eintragung ist das Stellen eines Antrags. Dieser Antrag muß im System registriert werden. Weiterhin sind zum Nachweis eines Rechts Urkunden notwendig, die das System ebenfalls registriert.
Ausgehend von den vorangegangenen Überlegungen werden nun die elementaren Operationen mit ihren Vorbedingungen und Wirkungen beschrieben.
1. Externe Operationen

- *Dokumente* müssen als Beweise dem Grundbuchamt vorgelegt und von diesem *registriert* werden.
  Vorbedingungen: Ein Dokument darf nicht mehrfach registriert werden, um die Eindeutigkeit zu gewährleisten.
  Wirkungen: Ein Dokument wird dem Grundbuchsystem hinzugefügt.
- Ein *Antrag* muß gestellt werden, damit ein Grundbuchverfahren eingeleitet werden kann. Dieser Antrag wird vom Grundbuchamt *registriert*.
  Vorbedingungen: Es darf noch kein Grundbuchverfahren laufen, das dasselbe Grundstück betrifft.
  Wirkungen: Ein Antrag wird dem Grundbuchsystem hinzugefügt.

2. Interne Operationen:

- Die *Eintragung eines Vollrechts* an einem Grundstück, also eines Eigentumsrechts, wird im Grundbuch *eingetragen*.
  Vorbedingungen: Das Vollrecht darf noch nicht eingetragen sein.

- Wirkungen: Das Eigentumsrecht wird zu Gunsten des Käufers im Grundbuch eingetragen. Eine auf dem Grundstück lastende Hypothek bleibt davon unberührt und besteht weiter fort.
- Ein Teilrecht, das den Wiederverkaufswert eines Grundstücks einschränkt, kann in das Grundbuch *eingetragen* werden. In diesem Modell handelt es sich hierbei um eine Hypothek, die einen Wert hat, der mit ihr eingetragen wird.
  Vorbedingungen: Das Recht darf im Grundbuch noch nicht eingetragen sein.
  Wirkungen: Zu Gunsten des Gläubigers wird die Hypothek dann als Recht an dem Grundstück eingetragen. Weiterhin wird der Wert der Hypothek eingetragen.
- Eine Teilrecht, das den Wiederverkaufswert eines Grundstücks einschränkt, kann *gelöscht* werden. In diesem Modell handelt es sich hierbei um eine Hypothek.
  Vorbedingungen: Das zu löschende Recht muß zusammen mit dem korrekten Wert im Grundbuch eingetragen sein.
  Wirkungen: Daraufhin wird die Hypothek als eingetragenes Recht zusammen mit ihrem Wert gelöscht.
- Ein Teilrecht, das den Nutzungswert eines Grundstücks einschränkt, kann *eingetragen* werden. Dies ist in diesem Modell ein Nießbrauch.
  Vorbedingungen: Das Recht darf noch nicht eingetragen sein.
  Wirkungen: Der Nießbrauch wird als Recht im Grundbuch eingetragen.
- Ein Teilrecht, das den Nutzungswert eines Grundstücks einschränkt, also zum Beispiel ein Nießbrauch, kann *gelöscht* werden.
  Vorbedingungen: Das zu löschende Recht muß im Grundbuch eingetragen sein.
  Wirkungen: Der Nießbrauch wird als Recht im Grundbuch gelöscht.
- Die *Vereinigung* von *Grundstücken* kann in das Grundbuch *eingetragen* werden.
  Vorbedingungen: Die beiden Grundstücke, die vereinigt werden sollen, müssen im Grundbuch eingetragen sein. Beide Grundstücke müssen denselben Eigentümer haben. Das bedeutet, daß unter der Vereinigung von Grundstücken die Vereinigung im eigenen Besitz verstanden wird.
  Wirkungen: Eines der Teilgrundstücke hört auf zu bestehen. Es wird Bestandteil des neuen Grundstücks. Die Flurstücke, aus denen die Einzelgrundstücke bestanden, werden dem gebildeten Grundstück zugeordnet. Rechte die an dem Grundstück bestanden, das im Zuge der Vereinigung seine Selbständigkeit verliert, werden auf das Gesamtgrundstück übertragen.
- Die *Teilung* von *Grundstücken* kann in das Grundbuch *eingetragen* werden.
  Vorbedingungen: das Grundstück, das geteilt werden soll, muß im Grundstücksverzeichnis vorhanden sein. Das neu entstehende Grundstück darf noch nicht vorhanden sein. Es muß ein Flurstück angegeben werden, daß von dem Ausgangsgrundstück getrennt werden soll.
  Wirkungen: Ein neues Grundstück wird in das Grundbuch eingetragen. Es besteht aus dem abgeteilten Flurstück und das andere, das Ausgangsgrundstück, aus den verbleibenden Flurstücken. Eine auf dem Ausgangsgrundstück lastende Hypothek besteht an ihm fort.

3.6.4 Komplexe Abläufe

Zur Erreichung einer Eintragung im Grundbuch sind sowohl die Einhaltung verschiedener Verfahrensvoraussetzungen (z.B. die Voreintragung des Antragstellers bei einer Grundstücksteilung) als auch formeller Voraussetzungen (z.B. das Stellen eines Antrags) notwendig. Die externen Operationen schaffen die formellen Voraussetzungen für die internen Operationen. Wie wirken nun aber interne und externe Operationen zusammen? Wie sieht die Verfahrensweise des Grundbuchamtes aus? Dies wird im folgenden beschrieben.
Zu jeder der elementaren internen Operationen wird nun ein komplexer Ablauf eingeführt, der die Arbeitsweise des Grundbuchamts zur Erreichung eines Eintrags im Grundbuch

repräsentiert. Komplexe Abläufe im Grundbuchsystem bilden Grundbuchverfahren ab. Sie sind also Teil des Verfahrensrechts. Grundbuchverfahren sind gerichtet auf die Erreichung der Eintragung eines bestimmten Rechts, also auf die Eintragung einer materiellen Rechtsänderung. Zur Einleitung eines Grundbuchverfahrens sind bestimmte Voraussetzungen notwendig, die als Ausgangspunkt beschrieben werden.

3.6.4.1 Verfahrensvoraussetzungen

Um ein Grundbuchverfahren zu eröffnen, ist ein Antrag notwendig. Ist dieser beim Grundbuchamt vorgelegt, sind die Verfahrensvoraussetzungen zu prüfen, die zur Eröffnung eines Grundbuchverfahrens erfüllt sein müssen. Sie gliedern sich in die allgemeinen und die speziellen Verfahrensvoraussetzungen. Im Grundbuchrecht gehört zu den allgemeinen Verfahrensvoraussetzungen, daß das Grundbuchamt zuständig ist. Ein Grundbuchamt ist dann zuständig, wenn das oder die das Verfahren betreffenden Grundstücke in dem ihm zugeordneten Grundbuch registriert sind. Zu den speziellen Voraussetzungen gehört die Antragsberechtigung der Person, die den Antrag stellt. Eine Person ist antragsberechtigt, wenn sie entweder von der Eintragung betroffen ist oder von ihr begünstigt wird. Im Falle eines Eigentumstransfers wären beispielsweise sowohl Käufer als auch Verkäufer antragsberechtigt. Diese Zuständigkeit des Grundbuchamts und die Antragsberechtigung werden in diesem Modell als hinreichend zur Eröffnung eines Grundbuchverfahrens angenommen.

3.6.4.2 Grundbuchverfahren

Das Grundbuchverfahren schafft die Verbindung von materiellem Recht und seiner Abbildung im Grundbuch. Entscheidender Punkt ist die Prüfung der Voraussetzungen für die Eintragung eines Rechts im Grundbuch. Ein Antrag beim Grundbuchamt setzt ein Grundbuchverfahren in Gang. Zuerst sind die Verfahrensvoraussetzungen zu prüfen, sind sie erfüllt, kann das Verfahren beginnen. In Abhängigkeit der Art des Antrags werden unterschiedliche Eintragungen im Grundbuch angestrebt. Es sind die folgenden Verfahren möglich:

- Eintragung einer Grundstücksteilung
  Soll eine Grundstücksteilung vorgenommen werden, dann ist nach Prüfung der Verfahrensvoraussetzungen das Grundbuchverfahren zu eröffnen. Es wird geprüft, ob ein ordnungsgemäßer Antrag vorliegt. Wenn es sich um ein zusammengesetztes Grundstück handelt und das abzuteilende Flurstück Bestandteil des zu teilenden Grundstücks ist, dann kann die Grundstücksteilung vorgenommen werden. Nach der Grundstücksteilung bestehen Rechte, die am Gesamtgrundstück bestanden, an den Einzelgrundstücken fort.
- Eintragung einer Grundstücksvereinigung
  Soll eine Grundstücksvereinigung vorgenommen werden, dann ist nach Prüfung der Verfahrensvoraussetzungen das Grundbuchverfahren zu eröffnen. Es wird geprüft, ob ein ordnungsgemäßer Antrag vorliegt. Danach kann die Vereinigung vorgenommen werden. Rechte, die an den Einzelgrundstücken bestanden, werden auf das neue Gesamtgrundstück übertragen.
- Eintragung eines Eigentumstransfers
  Soll ein Eigentumstransfer stattfinden, dann wird das Grundbuchverfahren eingeleitet, wenn die Verfahrensvoraussetzungen erfüllt sind. Es wird geprüft, ob ein ordnungsgemäßer Antrag vorliegt. Das Beweisdokument über den Übergang des Rechts, also der Kaufvertrag, muß dem Grundbuchamt vorliegen. Die Person, die in diesem Dokument als Verkäufer genannt wird, muß zum Zeitpunkt des Verkaufs der Eigentümer des Grundstücks sein.
  Sind diese Bedingungen erfüllt, ist der Eintragungsprozeß zu vollziehen und die Eintragung des Eigentumstransfers an dem Grundstück vorzunehmen. Die Eintragung des

Eigentumstransfers erfolgt durch Löschung des alten und Eintragung des neuen Eigentümers.

- Eintragung einer Hypothek
  Soll eine Hypothek eingetragen werden, dann ist das Grundbuchverfahren zu eröffnen, wenn die Verfahrensvoraussetzungen erfüllt sind. Es wird geprüft, ob dem Grundbuchamt ein ordnungsgemäßer Antrag vorliegt. Das Beweisdokument muß vorgelegt werden. Das Grundstück, das belastet werden soll, muß der Person gehören, die im Beweisdokument als der Empfänger der Hypothek eingetragen ist. Der Wert der Hypothek muß größer als Null sein.
  Wenn diese Bedingungen erfüllt sind, kann die Hypothek eingetragen werden.
- Löschung einer Hypothek
  Soll eine Hypothek gelöscht werden, dann beginnt das Grundbuchverfahren bei Erfüllung der Verfahrensvoraussetzungen. Es wird geprüft, ob ein Antrag vorliegt. Die Löschung muß mit dem Eigentümer der Hypothek vertraglich vereinbart sein. Dieser Vertrag muß als Beweis vorgelegt werden.
  Daraufhin kann die Löschung vollzogen werden.
- Eintragung eines Nießbrauchs
  Soll eine Nießbrauch eingetragen werden, dann ist das Grundbuchverfahren zu eröffnen, wenn die Verfahrensvoraussetzungen erfüllt sind. Es wird geprüft, ob dem Grundbuchamt ein ordnungsgemäßer Antrag vorliegt. Ein Beweisdokument muß vorgelegt werden. Das Grundstück, das mit dem Nießbrauch belastet werden soll, muß der im Beweisdokument genannten Person gehören.
  Danach kann der Nießbrauch eingetragen werden.

3.7 Die Beziehung von materiellem Recht und Grundbuch

Die internen Operationen des Grundbuchsystems sind analog den Aktionen im Modell der materiellen Rechtsverhältnisse. Dies muß auch so sein, sind doch die internen Operationen im Grundbuch Abbildungen der Veränderungen der materiellen Rechtsverhältnisse. Veränderungen in den materiellen Rechtsverhältnissen betreffen nicht direkt ihre Abbildung im Grundbuch. Die Verbindung zwischen materiellem Recht und seiner Abbildung im Grundbuch wird durch externe Operationen geschaffen. Externe Operationen bringen die Beweise über das Stattfinden von materiellem Rechtsänderungen in das Grundbuchsystem ein. Diese Beweise begründen die Abbildung der materiellen Rechtsverhältnisse im Grundbuch. Damit sind zur Abbildung von materiellen Rechtsänderungen zwei Bestandteile notwendig: Erstens das Stattfinden dieser Veränderungen in der materiellen Welt und zweitens ihre Dokumentierung im Grundbuch. Die Dokumentierung erfolgt durch Einbringen der entsprechenden Beweisdokumente in das Grundbuchsystem. Welche Beziehung besteht nun zwischen den internen Operationen des Grundbuchs und den materiellen Rechtsänderungen? Jede materielle Rechtsänderung muß durch eine interne Operation des Grundbuchs abgebildet werden. Die Relationen, die über die materiellen Rechtsverhältnisse im Grundbuch Auskunft geben, müssen mit den entsprechenden Relationen im Modell der materiellen Rechtsverhältnisse in Übereinstimmung sein. Zum Beispiel muß, wenn im Modell der materiellen Welt eine Person Eigentümer eines Grundstücks ist, im Grundbuch ein Eigentumsrecht zu Gunsten dieser Person eingetragen sein. Ist diese Übereinstimmung nicht gegeben, stimmen materielles Recht und Grundbuch nicht überein und folglich ist das Grundbuch unrichtig. Diese Unrichtigkeit kann gekennzeichnet werden.

3.8 Die Berichtigung des unrichtigen Grundbuchs

In Kapitel 2 wurden verschiedene Möglichkeiten aufgeführt, wie das Grundbuch korrigiert werden kann, wenn es unrichtig ist. Prinzipiell kann die Unrichtigkeit darin bestehen, daß eine Eintragung überflüssig, inkorrekt oder nicht vorhanden ist. Inkorrekte und überflüssige

Eintragungen werden gelöscht, nicht vorhandene werden ergänzt. Prinzipiell ist das Grundbuchamt verpflichtet, die Richtigkeit des Grundbuchs zu erhalten. Aus rein praktischen Gründen, die der Erhaltung der Arbeitsfähigkeit des Grundbuchamts dienen, ist das Grundbuchamt aber nicht verpflichtet, selbst unrichtige Eintragungen zu suchen und zu erkennen. Wird dem Grundbuchamt eine Unrichtigkeit bekannt, so ist es allerdings verpflichtet, diese zu beseitigen. Um die Korrektur des Grundbuchs zu erreichen, ist es also notwendig, sie dem Grundbuchamt zur Kenntnis zu bringen. Dies soll in diesem Modell dadurch dargestellt werden, daß es möglich ist, eine Beschwerde gegen die Richtigkeit des Grundbuchs zu führen. Diese Beschwerde kann entweder die Löschung einer (inkorrekten bzw. überflüssigen) Eintragung zum Ziel haben oder die Nachholung einer fehlenden Eintragung.

3.9 Zusammenfassung

Inhalt dieses Kapitels war die Erstellung eines konzeptionellen Modells, das die wesentlichen Aspekte der Arbeit eines Grundbuchsystems abbildet. Dazu wurde das Verhältnis von materiellem Recht, Grundbuch und dem zu erstellenden Modell diskutiert. Der entscheidende Punkt ist, daß das Modell sowohl eine korrekte Abbildung der materiellen Rechtsverhältnisse als auch des formellen Rechts sein muß.
Das Modell beschreibt einerseits die Objekte und Relationen, also den Inhalt des Grundbuchsystems. Andererseits stellt es dar, wie sich die Objekte und Relationen verändern. Veränderungen werden beschrieben durch elementare Operationen und komplexe Abläufe. Elementare Operationen wurden danach unterschieden, ob sie die Abbildung des formellen oder des materiellen Rechts betreffen.
Externe Operationen bilden die Schnittstelle des Systems zur Außenwelt. Sie bewirken allerdings nur Veränderungen in der Abbildung des formellen Rechts. Dennoch schaffen sie die Voraussetzung für interne Operationen, die materielle Rechtsveränderungen abbilden. Interne und externe Operationen werden zusammengeführt durch Abläufe, die das formelle Recht, also das Verfahrensrecht abbilden.
Objekte im Modell des Grundbuchs sind nicht die Objekte der materiellen Welt, die Grundstücke, Personen und Rechte, sondern die Bezeichner dieser Objekte. Dies reduziert die Frage, wie Veränderungen der Objekte zu modellieren sind, auf die Frage, wie sich die Bezeichner der Objekte ändern.
Ausgehend von der hier erfolgten informellen Beschreibung, besteht nun der nächste Schritt darin, die Voraussetzungen für die Formalisierung zu schaffen. Das heißt, daß ein Ansatz entwickelt werden muß, der geeignet ist, das Modell abzubilden.

4 Formale Grundlagen

4.1 Einleitung

In diesem Kapitel werden die formalen Grundlagen eingeführt, um ein Grundbuchsystem zu repräsentieren. Hierbei wird ausgegangen von der Tatsache, daß zur Wissensrepräsentation zwei wichtige Schritte zu tun sind. Einerseits ist die Repräsentationssprache auszuwählen, und andererseits ist das Wissen in dieser Sprache auszudrücken. Dieses Kapitel widmet sich der Frage der Auswahl der Repräsentationssprache und ihrer formalen Beschreibung. Ausgegangen wird hierbei von den Anforderungen an die Repräsentation, die sich aus der Analyse des Grundbuchs und dem konzeptionellen Modell ergeben. Es wird motiviert, warum die Repräsentation in dem auf der Prädikatenlogik basierenden Situationskalkül vorteilhaft ist.
Die Repräsentation des Wissens wird als eine formale Spezifikation aufgefaßt. Ausgehend von dieser Überlegung wird gezeigt, wie der Situationskalkül zur Spezifikation der Evolution von Datenbanken genutzt werden kann. Der verwendete Ansatz bietet eine sehr geradlinige Beziehung zum logischen Programmieren. Prolog ist in diesem Bereich sehr verbreitet. Wissen kann in Prolog repräsentiert werden um eine ausführbare Spezifikation zu erstellen. Dies ist sowohl aus der Sicht der Spezifikation wie auch der Wissensrepräsentation sinnvoll, bietet dieser Prototyp doch die Möglichkeit, die Richtigkeit der Repräsentation durch Testen zu überprüfen.

4.2 Wissensrepräsentation

Wissensrepräsentation zielt darauf, Wissen in einer vom Computer zu verarbeitenden Form abzubilden (Russell und Norvig 1995, S.157). Sie besteht darin, die Objekte und Relationen aus dem betrachteten Gegenstandsbereich in einer geeigneten *Wissensrepräsentationssprache* zu formulieren und in einer *Wissensbasis* abzubilden. Hierbei ergeben sich die beiden wichtigen Punkte, die die Wissensrepräsentation bestimmen: Einerseits die Formulierung des Wissens, also der Bau der Wissensbasis, und andererseits die Wahl der Repräsentationssprache. Vor der Formulierung des Wissens steht die Wahl der Repräsentationssprache. Wie wird diese Wahl bestimmt? Sie wird zum einen davon bestimmt, welche Anforderungen die Wissensbasis erfüllen soll, und zum anderen von der Art des zu beschreibenden Problems. Anders gesagt: Was für Anforderungen stellt der Nutzer, und welche Anforderungen stellt der zu repräsentierende Bereich an die Wissensbasis?
Der Prozeß, eine Wissensbasis aufzubauen, wird *Knowledge Engineering* genannt (Russell und Norvig 1995, S. 217). Die Abbildung des Wissens in der Wissensbasis selbst ist nicht der Zweck des Wissensrepräsentation. Der Zweck ist, dieses Wissen verfügbar zu machen und aus dem vorhandenen Wissen neues Wissen abzuleiten. Die Wissensrepräsentationssprache muß also auch über die Fähigkeit verfügen, neue Fakten zu beweisen. Diese Gedanken sind Ausdruck des *deklarativen Ansatzes*. Er ist charakterisiert durch die folgenden Schritte (Russell und Norvig 1995, S.219):

1. Die Auswahl einer Wissensrepräsentationssprache,
2. den Aufbau der Wissensbasis,
3. der Ableitung neuer Fakten.

Der zentrale Punkt ist dabei, daß der *Knowledge Engineer* angibt, *was* für Wissen vorhanden ist, also was wahr oder falsch ist. Die Ableitungsprozedur übernimmt die Aufgabe, die vorhandenen Fakten zu einer Lösung des Problems zu bringen, beschreibt also *wie* bewiesen werden kann, was wahr oder falsch ist.

4.2.1 Anforderungen an die Wissensrepräsentationssprache

Diese Arbeit beinhaltet die Repräsentation eines Grundbuchsystems. Die Anforderungen des Gesetzgebers an das Grundbuch gelten natürlich auch für ein wissensbasiertes Grundbuchsystem. Zusammenfassend gesagt bestehen sie darin, die Rechtsverhältnisse an Grundstücken konsistent zu verwalten und darüber Auskunft zu geben (siehe Kapitel 2.2). Hieraus lassen sich zwei wichtige Anforderungen an die Repräsentationssprache ableiten.

1. Das Wissen muß konsistent und widerspruchsfrei verwaltet werden, und es muß die Möglichkeit bestehen, zu überprüfen, ob dies auch so ist.
2. Die Repräsentationssprache muß über den Inhalt der Wissensbasis Auskunft geben.

Welche Anforderungen an die Repräsentationssprache ergeben sich aus dem konkreten Bereich, der zu beschreiben ist? Anders gesagt: Welche Ansprüche ergeben sich aus der Tatsache, daß ein Grundbuchsystem modelliert werden soll? Zwei zentrale Probleme haben sich aus der Analyse des Grundbuchs ergeben, die weitere Anforderungen an die Repräsentationssprache stellen:

3. Das Grundbuchsystem muß unter der Bedingung ständiger Veränderungen beschrieben werden.
4. Inkonsistenzen und die Probleme, die sich aus der Unvollständigkeit der Repräsentation ergeben, müssen behandelt werden.

Diese vier Punkte definieren die Anforderungen an eine Repräsentationssprache, die geeignet ist, ein Grundbuchsystem abzubilden.

4.2.2 Warum Prädikatenlogik?

Warum stellt eine auf der Prädikatenlogik basierende Sprache eine geeignete Grundlage zur Repräsentation dar? An den Anfang soll gestellt werden, daß die Prädikatenlogik eine sehr wohlverstandene und weitverbreitete Sprache zur Wissensrepräsentation darstellt. Sie verbindet Syntax, also die Regeln, nach denen die Ausdrücke der Sprache gebildet werden, und die Bedeutung dieser Ausdrücke, die Semantik, in eindeutiger Weise. Die Beweistheorie schafft die Verbindung zwischen der Sprache und der Bedeutung der Sprache in der Welt. Sie legt fest, wie die Regeln aussehen, nach denen aus Ausdrücken neue Ausdrücke geformt werden. Damit verbunden sind die Begriffe Korrektheit und Vollständigkeit, die besagen, daß einerseits aus der Repräsentation keine bezüglich der modellierten Welt falschen Ausdrücke abgeleitet werden können und andererseits auch alles abgeleitet werden kann, was der Bedeutung entspricht. Aus dieser kurzen (informellen) Schilderung ergeben sich bezüglich der oben angegebenen Anforderungen folgende Vorteile: Prädikatenlogik bildet eine eindeutige und konsistente Basis zur Repräsentation von Wissen. Der Inhalt der Wissensbasis läßt sich auf Basis der Beweistheorie abfragen: Wenn etwas in der Wissensbasis gültig ist, dann läßt sich ein Beweis dafür konstruieren. Für eine Einführung in die Prädikatenlogik wird auf (Rogers 1971) verwiesen.

4.2.3 Warum Situationskalkül?

Prädikatenlogik bildet konsistent ab, *was* in der modellierten Welt gültig ist. Keine Aussagen wurden bislang darüber getroffen, *wie* sich Veränderungen in der Welt modellieren lassen. Eng damit verbunden ist die Frage der *Kausalität:* Wie lassen sich die Konsequenzen von Veränderungen beschreiben? Dies ist der Ausgangspunkt für die Entwicklung des Situationskalküls (McCarthy und Hayes 1969) von John McCarthy. Der Grundgedanke ist folgender: Die Welt verändert sich infolge von Ereignissen oder Aktionen[1]. Den Zustand der Welt zu einem Zeitpunkt bestimmt die Situation, in der sie sich befindet. Aktionen bewirken den Übergang der Welt von einer Situation in die nächste. Auf diese Weise werden Veränderungen in der Welt ausschließliche als Folge von bestimmten Aktionen beschrieben.

[1] Der Bergriff Aktion wird in diesem Kapitel in Anlehnung an McCarthy verwendet. Er ist nicht gleichzusetzen mit dem Begriff der Aktion im Zusammenhang mit der Modellierung der materiellen Rechtsverhältnisse.

Situationen repräsentieren die Geschichte der Veränderungen in der Welt[2]. Veränderungen werden ausschließlich in den Wertänderungen bestimmter Prädikate abgebildet, den Fluents. Veränderungen beschränken sich also auf das Auftreten von Aktionen und der damit verbundenen Veränderung in den Wahrheitswerten der Fluents.
Ist dies ausreichend für die Beschreibung der Veränderungen in einem Grundbuchsystem? Intuitiv können die Aktionen mit den elementaren Operationen identifiziert werden. Elementare Operationen bewirken Eintragungen im Grundbuch oder ihre Löschung. Dies kann dadurch repräsentiert werden, daß einer Eintragung der Wert wahr bzw. falsch zugewiesen wird. Was gilt für komplexe Abläufe? McCarthy hat den Begriff der *Strategie* (McCarthy und Hayes 1969) eingeführt. Eine Strategie verbindet mehrere Aktionen, um ein bestimmtes Ziel zu erreichen. Dies korrespondiert zur Charakterisierung von Abläufen als Folge elementarer Operationen, die in Abschnitt 2.7.2.1 gegeben wurde. Damit hat der Situationskalkül die Fähigkeiten, die zur Repräsentation des Grundbuchs notwendig sind. Bemerkt sei noch, daß der Situationskalkül als prädikatenlogische Sprache vollständig die Vorteile der Prädikatenlogik übernimmt.

4.2.4 Reiters Erweiterung des Situationskalküls

Es hat sich herausgestellt, daß der Situationskalkül in der von McCarthy beschriebenen Form mit verschiedenen Problemen behaftet ist. Das bekannteste und auch schon von McCarthy erkannte, ist das *Frame Problem*. Es ist kein Problem des Situationskalküls allein, es ist ein Problem der Wissensrepräsentation allgemein. Es läßt sich wie folgt formulieren: Wie lassen sich die Bereiche der zu modellierenden Welt beschreiben, die sich nicht verändern? McCarthy schlug vor, neben der Verwendung von Effektaxiomen, die die Wirkung von Aktionen beschreiben, Frame Axiome zu nutzen, um explizit anzugeben, was sich nicht verändert (McCarthy und Hayes 1969). Dieser Gedanke ist insofern unpraktikabel, da unter Umständen sehr viele solche Axiome gebraucht werden, was die Komplexität stark erhöht. Deshalb steht am Anfang der Verwendung des Situationskalküls eine Lösung des Frame Problems. R. Reiter hat genau das getan, indem er ein Verfahren angibt, aus den Effektaxiomen eine Updatespezifikation zu gewinnen, die genau die Veränderungen und Invarianten angibt (Reiter 1991).
Ein zentraler Punkt bei der Repräsentation des Grundbuchsystems ist die Beschreibung von komplexen Abläufen. McCarthy hat mit der Einführung von Strategien den Weg gewiesen, wie sich komplexere Aktionen aus den elementaren Aktionen bilden lassen. Formal umgesetzt wurde dieser Gedanke in der *Programmiersprache* GOLOG (Levesque, Reiter et al. 1996). GOLOG bietet die Möglichkeiten, komplexe Abläufe zu definieren, wie sie für die Modellierung des Grundbuchsystems benötigt werden.
Eine Reihe weiterer Kritikpunkte am Situationskalkül sind bekannt geworden. Einige seien hier aufgezählt: Der Situationskalkül kann keine Zeit repräsentieren. Er kann keine Aktionen repräsentieren, die eine Dauer haben. Die indirekten Effekte von Aktionen lassen sich nicht beschreiben (das *Ramification Problem*), usw. Inhalt dieses Kapitels ist auch zu zeigen, wie diese Probleme gelöst werden können, wenn sie relevant für die Repräsentation eines Grundbuchsystems sind, bzw. warum sie nicht relevant sind für den hier zu beschreibenden Teil der Welt.
Der zentrale Punkt, der den Situationskalkül interessant macht für die Modellierung eines Grundbuchsystems, ist die Tatsache, daß er dazu genutzt werden kann, die Evolution von Datenbanken zu beschreiben. Anders gesagt kann die oben genannte Updatespezifikation als *Spezifikation von Datenbanktransaktionen* betrachtet werden (Reiter 1995).

[2] Die Auffassung, eine Situation repräsentiert eine Weltgeschichte, ist nicht auf McCarthy zurückzuführen, sondern stammt von Reiter. Bei McCarthy ist eine Situation ein Schnappschuß des Zustandes der Welt zu einem bestimmten Zeitpunkt.

In diesem Zusammenhang sind zwei Begriffe gefallen: *Spezifikation* und *Programmiersprache*. Es ist also erforderlich, den Zusammenhang von Wissensrepräsentation, Spezifikation und Programm zu betrachten.

4.3 Spezifikation

Spezifikation ist der Prozeß, ein System mit seinen erwünschten Eigenschaften zu beschreiben. Formale Spezifikation nutzt hierzu eine Sprache mit mathematisch definierter Syntax und Semantik (Clarke und Wing 1996). Formale Spezifikationen definieren die Eigenschaften eines Systems, bevor eine Implementation dieses Systems, also ein Programm erzeugt wird. Formale Spezifikationen präsentieren, *was* ein Programm tun soll, und nicht, *wie* es umgesetzt wird (Borgia, Mylopoulos et al. 1995). Das heißt, daß von der konkreten Umsetzung abstrahiert wird mit dem Zweck, nur die Eigenschaften festzulegen, die das Programm haben soll. Die dazu genutzte *formale Spezifikationssprache* muß, um für diesen Prozeß geeignet zu sein, folgende Eigenschaften haben (Borgia, Mylopoulos et al. 1995):

1. Sie muß syntaktisch geeignet sein. Das heißt, daß die Sprache in der Lage sein muß, die Eigenschaften eines Programms genau, einfach, verständlich, modular und in leicht veränderbarer Weise auszudrücken.
2. Sie muß die formale Behandlung unterstützen. Das heißt, die Sprache muß die Fähigkeit besitzen, Eigenschaften des Programms und seiner Entwicklung formal zu beweisen.

4.3.1 Formale Spezifikation und Wissensrepräsentation

In welchem Zusammenhang stehen Wissensrepräsentation und formale Spezifikation? Wissensrepräsentation schafft die Möglichkeiten, aus einer Wissensbasis neues Wissen zu gewinnen. Anders gesagt schafft sie die Voraussetzungen, um aus einer Problembeschreibung eine Lösung zu erhalten. Wichtig ist dabei der Punkt, daß von der Art der Gewinnung der Lösung abstrahiert wird, mit dem Zweck, den Nutzer der Wissensbasis von der Suche danach zu entlasten. Man kann sagen, daß es Ziel und auch Hauptvorteil der Wissensrepräsentation ist, von der konkreten Programmierung *weg* zu kommen, *hin* zu allgemeineren Methoden, Wissen auszudrücken. Es wird ausgedrückt, *was* das Problem und die Lösung ist und nicht, *wie* sie erreicht wird. Hierin liegt auch der Zusammenhang zur formalen Spezifikation. Formale Spezifikation zielt darauf, von der allgemeinen Problembeschreibung *hin* zu einem konkreten Programm zu kommen. Die Spezifikation beschreibt, *was* die Eigenschaften des Programms sein sollen, also was das Problem ist und was die Lösung. In diesem Sinn kann die Wissensrepräsentation als formale Spezifikation betrachtet werden. Die Spezifikation wird in Form einer Wissensbasis erstellt. Die Ableitungsmechanismen der Wissensbasis stellen die formalen Möglichkeiten bereit, die Eigenschaften des spezifizierten Programms zu beweisen. Aus den Anforderungen an die Spezifikationssprache ergibt sich die Wahl der geeigneten Wissensrepräsentationssprache.
Im Bezug auf das Grundbuchsystem bedeutet dies, daß die Repräsentation im Situationskalkül zu einer Spezifikation des Systems führt. Diese Spezifikation kann Grundlage für ein Programm, also eine Implementierung sein. Im konkreten Fall gilt es hierbei besonders, die Veränderungen des Systems, also elementare Operationen und Abläufe zu beschreiben. Man kann die auf dem Situationskalkül beruhende Spezifikation des Grundbuchsystems als Spezifikation einer Datenbank zusammen mit den die Datenbank verändernden Transaktionen auffassen, weiter noch als Spezifikation einer temporalen Datenbank, die Veränderungen in der Vergangenheit, wie auch der möglichen Zukunft abbildet.

4.3.2 Spezifikation von Datenbanktransaktionen

Aktionen bilden im Situationskalkül elementare Operationen ab. Man kann nun diese Aktionen als Spezifikation von Transaktionen zum Aktualisieren einer Datenbank betrachten. Mit dem Transaktionskonzept werden die möglichen Veränderungen in einer Datenbank auf eine endliche Anzahl von Aktualisierungsoperationen beschränkt. Das heißt, daß nicht

beliebige Daten hinzugefügt werden können, sondern nur solche, für die eine Transaktion definiert ist. Der einzige Weg, auf dem sich die Datenbank verändern kann, ist folglich der durch die benannten Transaktionen. Das Transaktionskonzept ist geeignet, das Konzept von elementaren Operationen im Grundbuch umzusetzen. Diese beschränken sich auf eine kleine Anzahl, deren Wirkungen genau festgelegt sind. Charakteristisch für das Transaktionskonzept ist die Angabe von Vor- und Nachbedingungen, die ausdrücken, was vor und nach der Ausführung der Transaktion gelten muß. Der Situationskalkül ist nun syntaktisch geeignet, da er, zusammen mit der Reiterschen Erweiterung (Reiter 1995), genau solche Vor- und Nachbedingungen in Gestalt der genannten Updatespezifikation definiert. Die Fluents des Situationskalküls werden als Relationen im Sinne der Datenbanktheorie aufgefaßt.
Bezüglich dieser Updatespezifikation kann nun bewiesen werden, welche Veränderungen sich ergeben, ob eine Transaktion überhaupt möglich ist und was die Datenbank nach einer solchen Folge von Transaktionen beinhaltet.

4.3.3 Spezifikation von komplexen Transaktionen

Am Anfang steht die Frage: Wie können komplexe Abläufe im Situationskalkül formuliert werden? Es wurde bereits erwähnt, daß Abläufe in GOLOG (Levesque, Reiter et al. 1996) beschrieben werden können. GOLOG stellt nun die geeignete Grundlage zur Spezifikation von komplexen Transaktionen dar. Komplexe Transaktionen sind dadurch charakterisiert, daß sie sich aus elementaren Transaktionen zusammensetzen und diese durch verschiedene Operatoren in Beziehung zueinander setzen. Zu diesen Operatoren gehören zum Beispiel Bedingungen, oder die Nacheinanderausführung von Transaktionen. GOLOG bietet die Möglichkeiten, komplexe Transaktionen aus einfachen zusammenzusetzen. Komplexe Transaktionen sind daher das geeignete Konzept um Abläufe zu beschreiben. Ein GOLOG Programm wird evaluiert bezüglich der zugrundeliegenden Spezifikation der elementaren Transaktionen. Das Ergebnis eines erfolgreichen Programmlaufs ist eine Folge von Transaktionen, die korrekt bezüglich dieser Spezifikation ist. Diese könnte Grundlage für die physische Umsetzung dieser Transaktionen sein, indem sie dann auf einer realen Datenbank ausgeführt wird. Es wird aus der Beschreibung der elementaren und komplexen Transaktionen, also der Beschreibung ihrer Eigenschaften, eine Folge von konkreten Transaktionen gewonnen, die diese Beschreibung umsetzen. Sie beschreiben einen Weg, wie diese komplexen Transaktionen realisiert werden, stellen also in diesem Sinn ein Programm dar, das die komplexen Transaktionen implementiert.

4.3.4 Ausführbare Spezifikationen und Implementation

Die Beschreibung komplexer Transaktionen als GOLOG Programme bildet die Grundlage für die Betrachtung als ausführbare Spezifikation. Warum soll eine Spezifikation ausführbar sein? Der Ausgangspunkt ist die Tatsache, daß überprüft werden muß, ob die Spezifikation die gewünschten Eigenschaften hat, die vom Nutzer erwartet werden. Nur dann kann auch das spezifizierte Programm diese gewünschten Eigenschaften haben. Eine ausführbare Spezifikation ermöglicht dies durch Testen, das heißt zu klären, ob sich die Spezifikation so verhält, wie es erwartet wird (Sanella 1988).
Warum kann nun die auf dem Situationskalkül basierende Spezifikation von Datenbankupdates als ausführbare Spezifikation betrachtet werden? Der Grund ist der, daß es sehr einfach ist, diese Spezifikation in der logischen Programmiersprache Prolog (Bratko 1990) zu formulieren. Diese Übertragung in Prolog Syntax ist korrekt bezüglich der ursprünglichen Spezifikation und kann als prototypische Implementation betrachtet werden, was in dieser Arbeit auch erfolgen wird.

4.4 Formale Grundlagen zur Repräsentation eines Grundbuchsystems - einleitende Gedanken

Im vorangegangenen Abschnitt wurde motiviert, warum eine auf dem Situationskalkül beruhende Repräsentation geeignet für die Beschreibung eines Grundbuchsystems ist. Zentraler Punkt ist hierbei, daß diese Repräsentation als Spezifikation des Systems betrachtet werden kann. Aktionen im Situationskalkül und GOLOG Programme spezifizieren Datenbanktransaktionen. In diesem Abschnitt werden diese Gedanken dadurch untermauert, daß die Repräsentationssprache formal eingeführt wird. Dieser Ansatz wird ergänzt um Konzepte, die notwendig zur Repräsentation des Grundbuchs sind, aber bisher nicht umgesetzt wurden. Das sind die Integrierung eines Zeitbegriffs, die Abbildung von Ereignissen, die eine Dauer haben, und die Aufnahme externer Ereignisse.
Wichtig ist für ein Grundbuchsystem, daß Anfragen gestellt werden können, das heißt, daß zu ermitteln ist, was in der Datenbank zu einem Zeitpunkt gilt. Dazu werden Verfahren vorgestellt, wie Anfragen an das System beantwortet werden können.

4.5 Die Formalisierung elementarer Transaktionen

Inhalt diese Abschnitts ist die Vorstellung der formalen Grundlagen, die zur Beschreibung elementarer Transaktionen notwendig sind.

4.5.1 Der Situationskalkül

Der Standardansatz in der Künstlichen Intelligenz zur Beschreibung dynamischer, sich verändernder Systeme ist der Situationskalkül (McCarthy und Hayes 1969). Dabei handelt es sich um eine prädikatenlogische Sprache erster Ordnung. Wenn die Welt sich in dem Zustand *s* befindet und *a* eine Aktion ist, dann wird der nachfolgende Zustand mit *do(a,s)* bezeichnet. Beispielsweise könnte *do(buy(X,Y,G),s)* die Situation bezeichnen, die entsteht, wenn eine Person X ein Grundstück G von einer Person Y kauft. Funktionssymbole drücken Aktionen im Situationskalkül aus. Alle Relationen, deren Wahrheitswerte sich verändern können im Ergebnis von Aktionen, bezeichnet man als *Fluents*. Diese werden beschrieben durch Prädikatensymbole, die einen Term der Sorte *Situations* als Parameter haben. Zum Beispiel könnte das Fluent *Owner(X,G,s)* bedeuten, daß X Eigentümer des Grundstücks G in der Situation s ist. Nun kann die Aktion *do(buy(X,Y,G),s)* natürlich nicht immer und von jedem ausgeführt werden. Auch hat sie nicht immer dieselben Wirkungen. Es ist die Einhaltung bestimmter Vorbedingungen notwendig. Zum Beispiel muß Y, um das Grundstück verkaufen zu können, Eigentümer des Grundstücks G sein. Weiterhin ist es erforderlich, die Auswirkungen von Aktionen auf die Welt zu beschreiben. Dies erfolgt durch die Angabe von Effektaxiomen, die für eine gegebene Aktion die Veränderungen der Wahrheitswerte der Fluents spezifizieren. Entsprechend muß in unserem Beispiel *Owner(X,G,do(buy(X,Y,G),s))* gelten, als Ergebnis des Kaufs des Grundstücks G muß X Eigentümer dieses Grundstücks werden.
Notation: Im folgenden werden alle verwendeten Formeln als universal quantifiziert angenommen.

Beispiel 4.1 Ausschnitt aus einem Grundbuch

Ein Grundbuch verwaltet Rechte an Grundstücken und die sie beweisenden Dokumente. Dokumente können registriert werden. Rechte an Grundstücken können transferiert werden.

Fluents: *Dokument(g,p1,p2,r,s)*
Ein Dokument dient als Beweis des Übergangs eines Rechts zwischen zwei Personen an einem Grundstück.
Eingetragenes_Recht(g,p,r,s)
Ein Recht ist zu Gunsten einer Person an einem Grundstück eingetragen.

Aktionen: *register_Dokument(g,p1,p2,r)*
Ein Dokument wird vom Grundbuchamt als Beweismittel registriert.

transfer_Recht(g,p1,p2,r)
Das Recht r an einem Grundstück wird von einer Person zur anderen übertragen.

Effektaxiome: $\neg Dokument(g,p1,p2,r,s) \Rightarrow Dokument(g,p1,p2,r,do(register\_Dokument(g,p1,p2,r),s))$
Wenn ein Dokument noch nicht im Grundbuch ist, dann wird es registriert.
$Dokument(g,p1,p2,r,s) \wedge Eingetragenes\_Recht(g,p1,r,s) \Rightarrow$
$Eingetragenenes\_Recht(g,p2,r,do(transfer\_Recht(g,p1,p2,r),s))$
Wenn eine Person als Eigentümer eines Rechts an einem Grundstück im Grundbuch eingetragen ist und ein Dokument den Übergang des Rechts an eine andere Person beweist, dann wird dieses Recht zu Gunsten dieser Person im Grundbuch eingetragen.
$Dokument(g,p1,p2,r,s) \wedge Eingetragenes\_Recht(g,p1,r,s) \Rightarrow$
$\neg Eingetragenenes\_Recht(g,p1,r,do(transfer\_Recht(g,p1,p2,r),s))$
Das Recht des alten Eigentümers wird, nachdem er das Recht abgegeben hat, aus dem Grundbuch gelöscht.

4.5.2 Beschreibung von Datenbanktransaktionen

Die folgende Darstellung orientiert sich an (Reiter 1995). In diesem Artikel werden Ergebnisse aus früheren Veröffentlichungen zusammengefaßt (Reiter 1992b; Reiter 1992c; Reiter 1992a).
Die hier verwendete Version des Situationskalküls ist eine mehrsortierte prädikatenlogische Sprache zweiter Ordnung mit Gleichheit, die folgende Bestandteile hat:
Sie enthält endlich viele Funktionssymbole der Sorte *Transactions*. Diese bezeichnen die möglichen Transaktionen. Weiterhin enthält sie genau zwei Funktionsymbole der Sorte *Situations*: Die Konstante S_0 und die Funktion *do(a,s)*, wobei a der Sorte *Transactions* und s der Sorte *Situations* ist. S_0 ist die initiale Situation, also die Situation, in der sich das System befindet, bevor eine Transaktion ausgeführt wird. Die Funktion *do(a,s)* bezeichnet die Situation, die das Ergebnis der Transaktion a in Situation s ist. Das Prädikatensymbol *Poss(a,s)* hat die Bedeutung, daß in der Situation s die Transaktion a möglich ist. Das binäre Prädikat $<$ mit Argumenten der Sorte *Situations* hat für $s < s'$ die Interpretation, daß s' von s durch eine Folge von Transaktionen erreichbar ist. Die Sprache enthält weiterhin endlich viele Prädikatensymbole, wobei jedes unter seinen Argumenten genau eines der Sorte *Situations* hat. Diese werden als *Fluents* bezeichnet. Der Wahrheitsgehalt der *Fluents* kann sich in Abhängigkeit der aktuellen Situation verändern. Es wird angenommen, daß immer das letzte Argument der Sorte *Situations* ist. Außerdem kann die Sprache Prädikate und Funktionen enthalten, die situationsunabhängig sind, also keine Situationsargumente haben. Alle Objekte, die nicht der Sorte *Transactions* oder *Situations* sind, werden der Sorte *Objects* zugeordnet.
Nun folgen einige Definitionen, die im weiteren Verlauf wichtig sind.

Definition 4.1

Die einfachen Formeln sind die kleinste Menge für die gilt:

1. *$F(t_1,...,t_n,s)$ und $F(t_1,...,t_n,S_0)$ sind einfach, wenn F ein Fluent ist, $t_1,...,t_n$ Terme und s eine Variable der Sorte Situations ist.*
2. *Alle Gleichheitsatome sind einfach.*
3. *Alle anderen Atome mit Prädikatensymbolen außer Poss und $<$ sind einfach.*
4. *Wenn P_1 und P_2 einfach sind, dann sind es auch $\neg P_1$, $P_1 \wedge P_2$, $P_1 \vee P_2$, $P_1 \Rightarrow P_2$, $P_1 \Leftrightarrow P_2$.*
5. *Wenn P einfach ist, so ist es auch $\exists x\ P$ und $\forall x\ P$, wenn x eine Individuenvariable nicht der Sorte Situations ist.*

Die einfachen Formeln sind also die Formeln, in denen nicht *Poss* und $<$ vorkommen, in deren Fluents nicht das Symbol *do* vorkommt und in denen nicht über Situationen quantifiziert wird.

Definition 4.2

Ein Transaction Precondition Axiom ist eine Formel der Form

$$\forall x_1,...,x_n,s\ Poss(T(x_1,...,x_n),s) \Leftrightarrow \Pi_T(x_1,...,x_n,s) \tag{4.1}$$

T ist eine n-stellige Funktion der Sorte *Transactions* und Π_T eine einfache Formel, deren freie Variablen aus $x_1,...,x_n,s$ sind.

Ein Transaction Precondition Axiom legt die Vorbedingung für eine Transaktion fest.

Definition 4.3

Ein Successor State Axiom (für ein n+1-stelliges Fluent F) ist eine Formel der Form:

$$\forall a,s\ Poss(a,s) \Rightarrow \forall x_1,...,x_n\ (F(x_1,...,x_n,do(a,s)) \Leftrightarrow \Phi_F(x_1,...,x_n,a,s)) \tag{4.2}$$

Dabei ist Φ_F eine einfache Formel, deren freie Variablen aus $x_1,...,x_n,a,s$ sind.

Ein Successor State Axiom legt die Wirkung der Ausführung einer Transaktion auf ein Fluent fest.

Definition 4.4

Eine Datenbank ist eine Menge von Formeln der folgenden Form:

$$D=\Sigma \cup D_{ss} \cup D_{tp} \cup D_{unt} \cup D_{uns} \cup D_{S0} \tag{4.3}$$

mit:

- Σ ist die folgende Menge der Axiome über Situationen:

$$\forall s\ \neg s < S_0 \tag{4.4}$$

$$\forall a,s,s'\ s < do(a,s') \Leftrightarrow Poss(a,s') \wedge s \leq s' \tag{4.5}$$

$$\forall P(P(S_0) \wedge \forall a,s\ (P(s) \Rightarrow P(do(a,s))) \Rightarrow \forall s\ P(s)) \tag{4.6}$$

  wobei $s \leq s'$ eine Abkürzung für $s<s' \vee s=s'$ ist.
- *D_{SS} ist die Menge der Successor State Axiome, für jedes Fluent genau eins.*
- *D_{tp} ist die Menge der Transaction Precondition Axiome, für jede Transaktion genau eins.*
- *D_{unt} ist die Menge der Unique Names Axiome für Transaktionen: Für verschiedene Transaktionen T und T' gilt:*

  $$T(x_1,...,x_n) \neq T'(y_1,...,y_m)$$

  und identische Transaktionen haben identische Argumente:

  $$T(x_1,...,x_n)=T(y_1,...,y_m) \Rightarrow x_1=y_1 \wedge ... \wedge x_n=y_n$$
- *D_{uns} ist die Menge der Unique Names Axiome für Situationen:*

  $$\forall a,s\ S_0 \neq do(a,s)$$

  $$\forall a,a',s,s'\ do(a,s)=do(a',s') \Rightarrow a=a' \wedge s=s'$$
- *D_{S0} ist eine Menge von prädikatenlogischen Formeln erster Ordnung mit der Eigenschaft, daß S_0 der einzige Term der Sorte Situations ist, der in den Fluents dieser Formeln vorkommt.*

(4.4) und *(4.5)* definieren induktiv die Ordnungsrelation über Situationen. *(4.6)* ist ein Induktionsaxiom zweiter Ordnung, das den Bereich der Situationen auf die initiale Situation

S_0 und ihre Nachfolgesituationen einschränkt. *(4.6)* ist die einzige Formel zweiter Ordnung, alle anderen Formeln in der Theorie sind erster Ordnung. D_{S0} ist die initiale Datenbank, also die Datenbank bevor eine Transaktion stattgefunden hat.

Notation: $do([a_1,...,a_n],s)$. Wenn $a_1,...,a_n$ Terme der Sorte *Transactions* sind, dann wird definiert:

$$do([],s) \stackrel{def}{=} s$$

$$do([a_1,...,a_n],s) \stackrel{def}{=} do(a_n,do(do([a_1,...,a_{n-1}],s)))$$

$do([a_1,...,a_n],s)$ ist also Abkürzung für $do(a_n,do(a_{n-1},...,do(a_1,s)...))$ und bezeichnet die Situation, die entsteht, wenn in Situation s die Transaktion a_1, dann die Transaktion a_2, usw. ausgeführt werden.

Beispiel 4.1 Ausschnitt aus einem Grundbuch (fortgesetzt)
Fluents: *Dokument(g,p1,p2,r,s)*
Eingetragenes_Recht(g,p,r,s)
Transaktionen: *register_Dokument(g,p1,p2,r)*
transfer_Recht(g,p1,p2,r)
Transaction Precondition Axiome:
Poss(register_Dokument(g,p1,p2,r),s)⇔¬Dokument(g,p1,p2,r,s)
Poss(transfer_Recht(g,p1,p2,r),s)⇔ Dokument(g,p1,p2,r,s)∧Eingetragenes_Recht(g,p1,r,s)
Successor State Axiome
Poss(a,s)⇒(Eingetragenenes_Recht(g,p2,r,do(a,s))⇔
a=transfer_Recht(g,p1,p2,r)∨Eingetragenenes_Recht(g,p2,r,s)∧
¬∃p3(a=transfer_Recht(g,p2,p3,r)))
Poss(a,s)⇒(Dokument(g,p1,p2,r,do(a,s))⇔
a=register_Dokument(g,p1,p2,r)∨Dokument(g,p1,p2,r,s))

Beispiel 4.1 liegt nun in zwei Versionen vor. Im ersten Fall werden die Veränderungen im Grundbuch durch Effektaxiome abgebildet, im zweiten Fall durch eine Updatespezifikation, bestehend aus Vorbedingungen und Nachbedingungen. Wie kann nun die zweite Version aus der ersten gewonnen werden?

4.5.3 Die Gewinnung der Updatespezifikation

In welchem Verhältnis stehen die ursprüngliche Version des Situationskalküls und die Datenbankspezifikation, wie sie oben angegeben wurde? Hierbei ist der entscheidende Punkt, daß die zweite Variante eine Lösung des Frame Problems darstellt. Diese Lösung kann aus den Effektaxiomen durch das Verfahren gewonnen werden, das im folgenden beschrieben wird.
Ausgegangen wird von dem folgenden Ausschnitt aus Beispiel 4.1:

Dokument(g,p1,p2,r,s)∧Eingetragenes_Recht(g,p1,r,s)⇒
Eingetragenenes_Recht(g,p2,r,do(transfer_Recht(g,p1,p2,r),s))

Dies ist das positive Effektaxiom, das die Wirkung der Transaktion *transfer_Recht* auf das Fluent *Eingetragenes_Recht* beschreibt. Dies kann nun äquivalent geschrieben werden als:

Dokument(g,p1,p2,r,s)∧Eingetragenes_Recht(g,p1,r,s)∧a=transfer_Recht(g,p1,p2,r)⇒
Eingetragenenes_Recht(g,p2,r,do(a,s))

Weiterhin wird das Prädikat Poss(a,s) eingeführt:

Dokument(g,p1,p2,r,s)∧Eingetragenes_Recht(g,p1,r,s)⇒ Poss(transfer_Recht(g,p1,p2),s)

Diese Formel bezeichnet die Vorbedingung für die Transaktion a.
Poss(a,s) sagt aus, ob die Aktion a in Situation s möglich, also die Vorbedingung erfüllt ist.

Dadurch kann das Effektaxiom wie folgt geschrieben werden

$$Poss(a,s) \wedge a=transfer\_Recht(g,p1,p2,r) \Rightarrow Eingetragenenes\_Recht(g,p2,r,do(a,s))$$

Diese Formel wird bezeichnet als allgemeines positives Effektaxiom für das Fluent *Eingetragenes_Recht*.

Das obige Beispiel kann nun verallgemeinert werden:
Für jedes Fluent R sind ein positives und ein negatives allgemeines Effektaxiom gegeben:

$$Poss(a,s) \wedge y^{+}(a,s) \Rightarrow R(do(a,s)) \tag{4.7}$$

$$Poss(a,s) \wedge y^{-}(a,s) \Rightarrow \neg R(do(a,s)) \tag{4.8}$$

Für jede Transaktion A ist die Vorbedingung in Form eines Transaction Precondition Axioms definiert:

$$\Pi_{A}(s) \Rightarrow Poss(A,s)$$

$y^{+}(a,s)$ bzw. $y^{-}(a,s)$ gibt jeweils eine Fluent Vorbedingung an, das heißt, sie bestimmt unter welchen Bedingungen durch eine Transaktion a das Fluent R wahr bzw. falsch wird.

Es wird nun die folgende *Vollständigkeitsannahme* gemacht:

> *Die positiven und negativen Effektaxiome für ein Fluent R geben alle Bedingungen an, die dazu führen, daß R wahr bzw. falsch wird in der Folgesituation.*

Daraus folgt, daß wenn eine Transaktion in einer Situation s möglich ist und ein Fluent R seinen Wert in Folge dieser Transaktion von falsch auf wahr ändert, so muß $y^{+}(a,s)$ gegolten haben in Situation s.
Dieser Gedanke wird formal durch die folgenden *Explanation Closure Axiome (Schubert 1990)* ausgedrückt:

$$Poss(a,s) \wedge R(s) \wedge \neg R(do(a,s)) \Rightarrow y^{-}(a,s) \tag{4.9}$$

$$Poss(a,s) \wedge \neg R(s) \wedge R(do(a,s)) \Rightarrow y^{+}(a,s) \tag{4.10}$$

Dies führt zu dem folgenden Ergebnis (Reiter 1991, Result 1):
Die allgemeinen Effektaxiome (4.7) und (4.8), zusammen mit den Explanation Closure Axiomen (4.9) und (4.10) sind logisch äquivalent zu der folgenden Formel:

$$Poss(a,s) \Rightarrow (R(do(a,s)) \Leftrightarrow y^{+}(a,s) \vee R(s) \wedge \neg y^{-}(a,s)) \tag{4.11}$$

Dies ist ein Successor State Axiom der Form *(4.2)*.
Die obige Äquivalenz gilt allerdings nur unter der Konsistenzannahme, daß $\neg(y^{+}(a,s) \wedge y^{-}(a,s))$ gilt. Unter dieser Bedingung ist es unmöglich R und ¬R gleichzeitig abzuleiten.
Für den obigen Ausschnitt aus Beispiel 4.1 läßt sich dann das folgende Successor State Axiom ableiten:

$$Poss(a,s) \Rightarrow (Eingetragenenes\_Recht(g,p2,r,do(a,s)) \Leftrightarrow$$
$$a=transfer\_Recht(g,p1,p2,r) \vee Eingetragenenes\_Recht(g,p2,r,s) \wedge$$
$$\neg \exists p3(a=transfer\_Recht(g,p2,p3,r)))$$

Dies entspricht genau dem erwarteten Resultat.

4.5.4 Das Frame Problem

Ein bei der Verwendung des Situationskalküls auftretendes Hauptproblem ist das Frame Problem. Es wurde bereits von John McCarthy selbst erkannt (McCarthy und Hayes 1969). Es besteht darin, in geeigneter Weise zu repräsentieren, daß Teile der Welt sich nicht

verändern. Im Falle des Situationskalküls bedeutet dies, daß bestimmte Fluents von einer Aktion unberührt bleiben. Warum die weiter oben beschriebene Updatespezifikation eine Lösung des Frame Problems darstellt, wird im folgenden (anhand von Beispiel 4.1) verdeutlicht.
Das Successor State Axiom für das Fluent *Eingetragenes_Recht* lautet:

$$Poss(a,s) \Rightarrow (Eingetragenenes\_Recht(g,p2,r,do(a,s)) \Leftrightarrow$$
$$a = transfer\_Recht(g,p1,p2,r) \vee Eingetragenenes\_Recht(g,p2,r,s) \wedge$$
$$\neg \exists p3(a = transfer\_Recht(g,p2,p3,r)))$$

Ziel ist es zu zeigen, daß der Wahrheitswert von einer beliebigen Transaktion unangetastet bleibt, die nichts mit dem Rechtstransfer zu tun hat. Was passiert also, wenn die Transaktion *register_Dokument* abläuft:

$$Poss(register\_Dokument(g,p1,p2,r),s) \Rightarrow$$
$$(Eingetragenenes\_Recht(g,p2,r,do(register\_Dokument(g,p1,p2,r),s)) \Leftrightarrow$$
$$register\_Dokument(g,p1,p2,r) = transfer\_Recht(g,p1,p2,r) \vee Eingetragenenes\_Recht(g,p2,r,s) \wedge$$
$$\neg \exists p3(register\_Dokument(g,p1,p2,r) = transfer\_Recht(g,p2,p3,r)))$$

Mit Unique Names Axioms *transfer_Recht* und *register_Dokument* erhält man:

$$Poss(register\_Dokument(g,p1,p2,r),s) \Rightarrow$$
$$(Eingetragenes\_Recht(g,p2,r,do(register\_Dokument(g,p1,p2,r)),s) \Leftrightarrow$$
$$FALSE \vee Eingetragenes\_Recht(g,p2,r,s) \wedge TRUE)$$

was äquivalent ist zu:

$$Poss(register\_Dokument(g,p1,p2,r),s) \Rightarrow$$
$$(Eingetragenes\_Recht(g,p2,r,do(register\_Dokument(g,p1,p2,r)),s) \Leftrightarrow Eingetragenes\_Recht(g,p2,r,s))$$

Das ist genau das, was erreicht werden sollte. Der Wahrheitswert von *Eingetragenes_Recht* bleibt unangetastet von einer beliebigen Transaktion, in diesem Fall *register_Dokument*, die nicht in der Updatespezifikation für dieses Fluent enthalten ist.

4.5.5 Wie sieht die Datenbank aus?

Wie sieht nun eine Datenbank aus, die der obigen Spezifikation entspricht? Die Spezifikation beschreibt das System bevor eine Transaktion ausgeführt wurde. Wenn sich das System verändert, wenn also Transaktionen ausgeführt werden, kommt die Folge der bisher ausgeführten Transaktionen zu der Spezifikation hinzu. Den Zustand der Datenbank bestimmt die Beschreibung des Anfangszustands und die Folge der ausgeführten Transaktionen. Wie kann nun aber der Zustand der Datenbank ermittelt werden? Anders formuliert: Was gilt in der Datenbank nach Ausführung einer Folge von Transaktionen?
Zunächst muß die Folge der Transaktionen gültig bezüglich der Spezifikation sein. Das heißt, daß für jede Transaktion in der Liste ihre Vorbedingungen wahr sein mußten in der Situation, in der sie ausgeführt wurde. Da in diesem Falle keine beliebigen Transaktionsfolgen betrachtet werden, sondern nur die, die bereits ausgeführt wurden, ist diese Bedingung erfüllt: Wenn eine Transaktion ausgeführt wurde, dann mußte natürlich ihre Vorbedingung erfüllt sein, da sie sonst nicht ausgeführt worden wäre. Wie kann nun aber entschieden werden, ob etwas gilt in einem Datenbankzustand? Die einzigen Veränderungen, die in der Datenbank vorkommen können, sind Veränderungen in den Wahrheitswerten der Fluents. Folglich muß ein Verfahren gefunden werden, das entscheidet, welche Wahrheitswerte die Fluents in einem Datenbankzustand haben.

4.5.6 Der Regressionsoperator R

Der Grundgedanke ist folgender: In einer Formel kommt ein Fluent $F(x_1,...,x_n,do(a,s))$ vor. Wenn das zugehörende Successor State Axiom die Form *(4.2)* hat, dann kann $F(x_1,...,x_n,do(a,s))$ durch die äquivalente Formel $\Phi_F(x_1,...,x_n,a,s)$, ersetzt werden. Diese nun verweist nur noch auf die Situation s, die die Vorgängersituation von *do(a,s)* ist. Auf diese Weise kann induktiv die Formel so verändert werden, daß sie nur noch auf die initiale Situation S_0 verweist. Diese

Formel ist äquivalent zur Ausgangsformel. Die Beantwortung der Frage, ob die Ausgangsformel gültig ist in einem Zustand, reduziert sich dann auf den Beweis der äquivalenten Formel bezüglich der initialen Datenbank.
Die hier intuitiv formulierte Idee wird als *Regression* (Waldinger 1977) bezeichnet. Regression wird der zentrale Mechanismus sein, um Formeln im Situationskalkül zu beweisen.

Um die formulierte Idee umzusetzen, muß die Ausgangsformel einigen syntaktischen Bedingungen genügen:

Definition 4.5

Für eine Formel A ist Regression anwendbar, wenn gilt:

1. *jeder Situationsterm hat die Form* $do([t_1,...,t_n],S_0)$ *für* $n \geq 0$ *und* $t_1,...,t_n$ *sind Terme der Sorte Transactions.*
2. *Für jedes Atom der Form Poss(a,s), das in A vorkommt, hat a die Form* $T(x_1,...,x_n)$*, wobei T ein n-stelliges Funktionssymbol der Sorte Transactions ist.*
3. *In A kommt das Symbol* $\leq$ *nicht vor.*

Für diese Klasse von Formeln wird der *Regressionsoperator* R wie folgt definiert:

Definition 4.6

Sei A eine Formel für die Regression anwendbar ist, dann ist der Regressionsoperator R bezüglich einer Datenbank D wie folgt definiert:

1. *Wenn A kein Fluent Atom ist, dann ist* $R[A]=A$.
2. *Wenn A ein Fluent Atom ist, dessen Situationsargument* S_0 *ist, dann ist* $R[A]=A$.
3. *Wenn A ein Fluent Atom der Form* $F(y_1,...,y_n,do(\alpha,\delta))$ *ist und das zugehörige Successor State Axiom hat die Form (4.2):*

   $$\forall a,s\ Poss(a,s) \Rightarrow (\forall x_1,...,x_n\ F(x_1,...,x_n,do(a,s)) \Leftrightarrow \Phi_F(x_1,...,x_n,a,s)).$$

   Dann ist $R[A]= R[\Phi_F(y_1,...,y_n,\alpha,\delta)]$.
4. *Wenn A ein Atom der Form* $Poss(\alpha(y_1,...,y_n),\delta)$ *ist, dann hat es ein Action Precondition Atom der Form (4.1):*

   $$\forall x_1,...,x_n,s\ Poss(T(x_1,...,x_n),s) \Leftrightarrow \Pi_T(x_1,...,x_n,s).$$

   Dann ist $R[A]=R[\Pi_\alpha(y_1,...,y_n,\delta)]$
5. *Wenn A eine Formel ist, dann ist* $R[\neg A]=\neg R[A]$ *und* $R[\forall v\ A]=\forall v\ R[A]$.
6. *Wenn* W_1 *und* W_2 *Formeln sind, dann ist* $R[W_1 \wedge W_2]= R[W_1] \wedge R[W_2]$.

R ersetzt alle Vorkommen von Fluents, die das Symbol *do* enthalten, durch Instanzen von Φ_F aus ihren Successor State Axioms. Es wird dadurch erreicht, daß die Verschachtelungstiefe der Funktion *do* um eins reduziert wird, da die Formeln Φ_F einfach sind und folglich die Funktion *do* nicht enthalten. Alle *Poss(a,s)* werden durch ihre Definitionen ersetzt.
Der Ersetzungsvorgang setzt sich nun fort bis keine Ersetzung mehr möglich ist. Dies ist dann der Fall, wenn S_0 der einzig vorkommende Situationsterm ist. Die so entstandene Formel ist äquivalent zur Ansgangsformel.
Folglich kann die Gültigkeit einer Formel bezüglich der initialen Datenbank entschieden werden. Dies spiegelt das folgende Theorem wider:

Theorem 4.1 (Regressionstheorem)

Wenn W eine Formel und D eine Datenbankspezifikation der Form (4.3) ist, dann gilt:

$$D \models W \Leftrightarrow D_{s0} \cup D_{una} \models R[W] \qquad (4.12)$$

Beispiel 4.1 (fortgesetzt) Die Anwendung des Regressionsoperators R

Die initiale Datenbank wird ergänzt um die Formel

Eingetragenenes_Recht(G,P1,Eigentum,S0).

In der initialen Situation ist ein Eigentumsrecht zu Gunsten der Person P1 am Grundstück G im Grundbuch eingetragen.

Es soll bereits die folgende Transaktion stattgefunden haben:

register_Dokument(G,P1,P2,Eigentum).

Es wurde ein Dokument registriert, das den Transfer des Eigentums am Grundstück G von Person P1 zu Person P2 beweist.

Dieser Eigentumsübergang soll im Grundbuch eingetragen werden. Dazu muß die Transaktion *transfer_Recht(G,P1,P2,Eigentum)* ausgeführt werden. Es wird nun mit Hilfe des Regressionsoperators R abgeleitet, daß diese Transaktion in der aktuellen Situation möglich ist. Formal ausgedrückt bedeutet dies, die Gültigkeit der Formel

$Poss(transfer\_Recht(G,P1,P2,Eigentum),do(register\_Dokument(G,P1,P2,Eigentum),S_0))$

zu beweisen. Dies geschieht durch Anwendung des Regressionsoperators R.

$R[Poss(transfer\_Recht(G,P1,P2,Eigentum),do(register\_Dokument(G,P1,P2,Eigentum),S_0))]$
$=$
$R[Dokument(G,P1,P2,Eigentum,do(register\_Dokument(G,P1,P2,Eigentum),S_0))\wedge$
$Eingetragenes\_Recht(G,P1,Eigentum,do(register\_Dokument(G,P1,P2,Eigentum),S_0))]$
$=$
$R[Dokument(G,P1,P2,Eigentum,do(register\_Dokument(G,P1,P2,Eigentum),S_0))]\wedge$
$R[Eingetragenes\_Recht(G,P1,Eigentum,do(register\_Dokument(G,P1,P2,Eigentum),S_0))]$
$=$
$(register\_Dokument(G,P1,P2,Eigentum)=register\_Dokument(G,P1,P2,Eigentum)\vee$
$Dokument(G,P1,P2,Eigentum,S_0))\wedge$
$(register\_Dokument(G,P1,P2,Eigentum)=transfer\_Recht(G,P1,P2,Eigentum)\vee$
$Eingetragenenes\_Recht(G,P1,Eigentum,S_0)\wedge$
$\neg\exists p3(register\_Dokument(G,P1,P2,Eigentum)=transfer\_Recht(G,P1,p3,Eigentum)))$

Mit Hilfe von Unique Names Axiomen für Transaktionen und der Tatsache, daß $Eingetragenenes\_Recht(G,P1,Eigentum,S_0)$ in der initialen Datenbank enthalten ist, ergibt sich aus dieser Formel:

$(TRUE\vee FALSE)\wedge$
$FALSE\vee(TRUE\wedge TRUE)$
$=$
$TRUE$

Das heißt, daß die Transaktion möglich ist.

4.6 Die Formalisierung komplexer Transaktionen

Neben der Beschreibung der elementaren Operationen im Grundbuchsystem muß die Formalisierung von Abläufen möglich sein. Abläufe sind dadurch gekennzeichnet, daß sie eine zeitliche Ausdehnung haben. Der beschriebene Ansatz muß also zuerst in diese Richtung erweitert werden. Dies erfolgt durch Einführung von Prozessen. Ausgehend davon werden dann komplexe Abläufe als komplexe Transaktionen beschrieben.

4.6.1 Prozesse im Situationskalkül

Ein Prozeß zeichnet sich dadurch aus, daß er eine zeitliche Ausdehnung hat. Da das System in der Zeit, die der Vorgang dauert, nicht blockiert sein soll, muß die zeitliche Überlappung von Prozessen möglich sein.

Prozesse können in der folgenden Weise abgebildet werden (Pinto und Reiter 1995):
Es werden hierfür zwei Elementartransaktionen eingeführt, die den Beginn und das Ende eines Prozesses markieren. Eine Transaktionen erzeugt den Prozeß, die andere terminiert ihn wieder. Zusätzlich muß zwischen diesen Transaktionen ersichtlich sein, daß ein Prozeß gerade läuft. Dazu wird ein weiteres Fluent eingeführt. In der Datenbankspezifikation werden für jede Transaktion, die ein Prozeß sein soll, folgende Veränderungen vorgenommen:

1. Ein Fluent *Laufender_Prozeß (...,s)* wird eingeführt.
2. Die ursprüngliche Transaktion *Prozeß(...)* wird ersetzt durch die Transaktionen *start_Prozeß(...)* und *end_Prozeß(...)*.
3. Transaction Precondition Axiome für *start_Prozeß(...)* und *end_Prozeß(...)* werden eingeführt in der Art, daß die Vorbedingung von *start_Prozeß(...)* denen von *Prozeß(...)* entspricht. Die Vorbedingung von *end_Prozeß(...)* enthält *Laufender_Prozeß (...,s),* ist also in jedem Fall nur als Abschluß eines Prozesses möglich.
4. Successor State Axiom für *Laufender_Prozeß (...,s):*
   $Poss(a,s) \Rightarrow (Laufender\_Prozeß(...,do(a,s)) \Leftrightarrow a=start\_Prozeß(s) \vee Laufender\_Prozeß(...,s) \wedge a \neq end\_Prozeß(...,s))$

Dies stellt sicher, daß das Prädikat *Laufender_Prozeß* nur zwischen den Start- und Endtransaktionen gilt.

Beispiel 4.1 Ausschnitt aus einem Grundbuch (fortgesetzt)

Das Beispiel wird erweitert um die Fähigkeit, Grundbuchverfahren abzubilden. Das Grundbuchverfahren wird als Prozeß modelliert. Eine Transaktion setzt es in Gang und eine beendet es wieder. Dazu werden die Punkte 1-4 abgearbeitet unter der Berücksichtigung der Tatsache, daß hier keine bestehende Transaktion als Prozeß modelliert, sondern ein neuer Prozeß der Spezifikation hinzugefügt wird.
1. Ein Fluent *Laufend_Grundbuchverfahren(g,s)* wird der Spezifikation hinzugefügt mit der Bedeutung, daß bezüglich des Grundstücks g ein Grundbuchverfahren läuft.
2. Zwei Transaktionen *start_Grundbuchverfahren(g)* und *end_Grundbuchverfahren(g)* werden der Spezifikation hinzugefügt, die den Prozeß Grundbuchverfahren einleiten und beenden.
3. Vorbedingungen für die Transaktionen werden definiert.

$$Poss(start\_Grundbuchverfahren(g),s) \Leftrightarrow \neg Laufend\_Grundbuchverfahren(g,s)$$

Ein Grundbuchverfahren an einem Grundstück soll möglich sein, wenn noch kein anderes läuft.

$$Poss(end\_Grundbuchverfahren(g),s) \Leftrightarrow Laufend\_Grundbuchverfahren(g,s)$$

Das Grundbuchverfahren kann beendet werden, wenn es überhaupt lief.
Diese beiden Bedingungen stellen sicher, daß nur ein Grundbuchverfahren an einem Grundstück zu einem Zeitpunkt stattfindet. Dies gewährleistet, daß keine Inkonsistenzen auf Grund interagierender Abläufe entstehen können.
4. Das Successor State Axiom für das Fluent *Laufend_Grundbuchverfahren(g,s)* wird definiert.

$$Poss(a,s) \Rightarrow (Laufend\_Grundbuchverfahren(do(a,s)) \Leftrightarrow a=start\_Grundbuchverfahren(g) \vee Laufend\_Grundbuchverfahren(g,s) \wedge \neg a=end\_Grundbuchverfahren(g))$$

4.6.2 Komplexe Transaktionen: GOLOG

In diesem Abschnitt werden die Grundlagen geschaffen, Abläufe als komplexe Transaktionen zu beschreiben.
Die Definition komplexer Transaktionen erfolgt wie in (Levesque, Reiter et al. 1996) als GOLOG[3] Prozeduren. Dies führt zu der folgenden Axiomatisierung.

[3] alGOL in LOGic

Zur Darstellung komplexer Transaktionen werden Ausdrücke eingeführt, die als Abkürzungen für logische Ausdrücke im Situationskalkül dienen. Komplexe Transaktionen können sein: elementare Transaktionen, Test Aktionen, Sequenzen von Transaktionen, die nichtdeterministische Auswahl einer von zwei Transaktionen oder eines Arguments für eine Transaktion. Wichtig ist auch die Definition von Iterationen und darauf aufbauend die Möglichkeit while-Schleifen zu formulieren. Auf der Basis dieser komplexen Transaktionsausdrücke erfolgt dann die Definition von Bedingungen und Prozeduren.
Zuerst wird das Symbol $Do(\alpha,s,s')$ eingeführt. Hierbei ist α ein Ausdruck für eine komplexe Transaktion. Intuitiv bedeutet dies, daß die Situation s' von s durch die Folge der durch α spezifizierten Transaktionen erreicht wird.
$Do(\alpha,s,s')$ ist induktiv definiert über die Struktur des ersten Arguments:

- Elementare Transaktionen:

$$Do(\alpha,s,s') \stackrel{def}{=} Poss(\alpha[s],s) \wedge s' = do(\alpha[s],s) \tag{4.13}$$

α bezeichnet eine Aktion, in der alle Situationsargumente in den funktionalen Fluents unterdrückt sind, die als Argumente in α vorkommen. α[s] bezeichnet eine Transaktion, in der diese Argumente wiederhergestellt sind.
Bemerkung: Dies ist allgemeiner als die Spezifikation verlangt: Da keine funktionalen Fluents zugelassen sind, müssen auch keine Situationsargumente wiederhergestellt werden.

- Testaktionen:

$$Do(?\varphi,s,s') \stackrel{def}{=} \varphi[s] \wedge s = s' \tag{4.14}$$

φ ist eine Formel aus dem Situationskalkül, in der alle Situationsargumente unterdrückt sind. φ[s] ist die Formel, die man aus φ erhält, wenn in allen Fluents die Situationsargumente wiederhergestellt werden (z.B.: φ ist $\forall x\ Grundstück(x)$, dann ist φ[s] $\forall x\ Grundstück(x,s)$).

- Sequenz:

$$Do(\alpha_1;\alpha_2,s,s') \stackrel{def}{=} \exists s^*\ Do(\alpha_1,s,s^*) \wedge Do(\alpha_2,s^*,s') \tag{4.15}$$

- Nichtdeterministische Auswahl zweier Transaktionen:

$$Do(\alpha_1 \mid \alpha_2,s,s') \stackrel{def}{=} Do(\alpha_1,s,s') \vee Do(\alpha_2,s,s') \tag{4.16}$$

- Nichtdeterministische Auswahl eines Transaktionsarguments:

$$Do((\pi x)\alpha(x),s,s') \stackrel{def}{=} \exists x\ Do(\alpha(x),s,s') \tag{4.17}$$

- Nichtdeterministische Iteration einer Transaktion α:

$$Do(\alpha^*,s,s') \stackrel{def}{=} \forall P\ (\forall s_1\ (P(s_1,s_1)) \wedge \forall s_1 \forall s_2 \forall s_3\ (P(s_1,s_2) \wedge Do(\alpha,s_2,s_3) \Rightarrow P(s_1,s_3))) \tag{4.18}$$

Transaktion α wird 0- oder mehrmals ausgeführt.

- Bedingungen:

$$\underline{if}\ \varphi\ \underline{then}\ \alpha_1\ \underline{else}\ \alpha_2\ \underline{endif} \stackrel{def}{=} (?\varphi;\alpha_1) \mid (?\neg\varphi;\alpha_2) \tag{4.19}$$

- Schleifen:

$$\underline{while}\ \varphi\ \underline{do}\ \alpha\ \underline{endwhile} \stackrel{def}{=} (?\varphi;\alpha)^*;?\neg\varphi \tag{4.20}$$

Wie wird nun solch ein komplexer Transaktionsausdruck ausgewertet? Die Transaktionsausdrücke werden durch die zugehörenden Formeln im Situationskalkül ersetzt.

Diese werden dann bezüglich der Datenbankspezifikation evaluiert. Das heißt, es wird versucht, folgenden Beweis zu führen:

$$Axioms \vdash \exists s\ Do(\alpha, S_0, s)$$

Um noch kompliziertere Transaktionen zu ermöglichen, erfolgt jetzt die Definition von Prozeduren.

- Prozeduren:

Für jedes Prädikatensymbol der Arität n+2, das ein Paar Situationsterme als Argumente hat, gilt:

$$Do(P(t_1,...,t_n),s,s') \stackrel{def}{=} P(t_1[s],...,t_n[s],s,s') \tag{4.21}$$

Dies soll als Prozeduraufruf dienen und bedeutet, daß die Ausführung dieser Prozedur mit den aktuellen Parametern $t_1,...,t_n$ eine Übergang von Situation s zu s' verursacht.
Bemerkung: Wieder ist die Definition allgemeiner als notwendig, da keine funktionalen Fluents zugelassen sind.

Die komplexen Transaktionen sind dann die Menge der Ausdrücke, die aus den atomaren Transaktionen und Prozeduraufrufen mit den obigen Konstruktionsregeln gebildet werden können.

- Programme:

Ein Programm besteht aus einer Reihe von Prozeduraufrufen P mit den Parametern $v_1,...,v_n$ und den Prozedurkörpern $\alpha_1,...,\alpha_n$, gefolgt von dem Hauptprogramm α_0. Hierbei sind alle α komplexe Transaktionen, wie oben beschrieben. Das Programm hat die Form:

$$\underline{proc}\ P_1(v_1)\alpha_1\ \underline{endProc}\ ;...;\ \underline{proc}\ P_n(v_n)\alpha_n\ \underline{endProc}\ ;\ \alpha_0$$

Dieses Programm wird wie folgt bezüglich der Spezifikation evaluiert:

$$Do(\{\underline{proc}\ P_1(v_1)\alpha_1\ \underline{endProc}\ ;...;\ \underline{proc}\ P_n(v_n)\alpha_n\ \underline{endProc}\ ;\ \alpha_0\},s,s') \stackrel{def}{=}$$

$$\forall P_1,...,P_n\left(\bigwedge_{i=1}^{n}(\forall s_1,s_2,v_i\ Do(\alpha_i,s_1,s_2) \Rightarrow P_i(v_i,s_1,s_2))\right) \Rightarrow Do(\alpha_0,s,s'))$$

Wenn alle Prozeduraufrufe des Programms mit den Parametern v_i abgeschlossen sind bezüglich der Evaluierung ihrer Prozedurkörper, dann bedeutet die Evaluierung des Hauptprogramms genau die Evaluierung des Gesamtprogramms. Anders ausgedrückt bedeutet diese Definition, daß unter der Voraussetzung, daß jede Evaluierung eines Prozedurkörpers einen bezüglich des Prozeduraufrufs gültigen Situationsübergang liefert, jede Auswertung des Hauptprogramms einen gültigen Situationsübergang für das Gesamtprogramm erzeugt. Die praktische Konsequenz aus dieser Definition ist, daß jede Auswertung des Hauptprogramms einen gültigen Programmlauf erzeugt.

Beispiel 4.1 Ausschnitt aus einem Grundbuch (fortgesetzt)

Die Eintragung eines Eigentumstransfers in einem Grundbuchverfahren.
Dazu wird das Grundbuchverfahren eröffnet. Als Beweis für den Eigentumstransfer muß ein Dokument (der Kaufvertrag) vorgelegt werden. Danach kann die Eintragung vorgenommen werden und das Grundbuchverfahren wird abgeschlossen.

Zur Modellierung dieses Ablaufs wird die folgende GOLOG Prozedur definiert:

proc *Eigentumstransfer (g,p,p',r)*
?(r=Eigentum);
start_Grundbuchverfahren(g);
register_Dokument(g,p',p,r);
transfer_Recht(g,p,p',r);
end_Grundbuchverfahren(g)
end proc

Diese Prozedur wird in folgender Weise in eine Formel im Situationskalkül übersetzt.

$Do([?(r=Eigentum);start\_Grundbuchverfahren(g);register\_Dokument(g,p',p,r);$
$transfer\_Recht(g,p,p',r);\ end\_Grundbuchverfahren(g)],s,s')$
$=$
$\exists s_1' .\ Do(?(r=Eigentum),s,s_1') \wedge Do([start\_Grundbuchverfahren(g);\ register\_Dokument(g,p',p,r);$
$transfer\_Recht(g,p,p',r);\ \ end\_Grundbuchverfahren(g)],s_1,s')$
$=$
$\exists s_1' .\ r=Eigentum \wedge s=s_1' \wedge$
$\exists s_1 .\ Do(start\_Grundbuchverfahren(g),s,s_1) \wedge Do([register\_Dokument(g,p',p,r);$
$transfer\_Eigentum(g,p,p');\ \ end\_Grundbuchverfahren(g)],s_1,s')$
$=$
$\exists s_1' .\ r=Eigentum \wedge s=s_1' \wedge$
$\exists s_1 .\ Do(start\_Grundbuchverfahren(g),s,s_1) \wedge \exists s_2 .\ Do(register\_Dokument(g,p',p,r),s_1,s_2) \wedge$
$Do([transfer\_Eigentum(g,p,p');\ \ end\_Grundbuchverfahren(g)],s_2,s')$
$=$
$\exists s_1' .\ r=Eigentum \wedge s=s_1' \wedge$
$\exists s_1 .\ Do(start\_Grundbuchverfahren(g),s,s_1) \wedge \exists s_2 .\ Do(register\_Dokument(g,p',p,r),s_1,s_2) \wedge$
$\exists s_3 .\ Do(transfer\_Recht(g,p,p',r),s_2,s_3) \wedge\ \ Do(end\_Grundbuchverfahren(g),s_3,s')$
$=$
$\exists s_1' .\ r=Eigentum \wedge s=s_1' \wedge$
$\exists s_1 .\ Poss(start\_Grundbuchverfahren(g),s) \wedge s_1=do(start\_Grundbuchverfahren(g),s) \wedge$
$\exists s_2 .\ Do(register\_Dokument(g,p',p,r),s_1,s_2) \wedge$
$\exists s_3 .\ Poss(transfer\_Recht(g,p,p',r),s_2) \wedge s_3=do(transfer\_Recht(g,p,p',r),s_2) \wedge$
$Poss(end\_Grundbuchverfahren(g,s_3) \wedge s'=do(end\_Grundbuchverfahren(g),s_3)$
$=$
$\exists s_1' .\ r=Eigentum \wedge s=s_1' \wedge$
$\exists s_1 .\ Poss(start\_Grundbuchverfahren(g),s) \wedge s_1=do(start\_Grundbuchverfahren(g),s) \wedge$
$\exists s_2 .\ Poss(register\_Dokument(g,p',p,r),s_1) \wedge s_2=do(register\_Dokument(g,p',p,r),s_1) \wedge$
$\exists s_3 .\ Poss(transfer\_Recht(g,p,p',r),s_2) \wedge s_3=do(transfer\_Recht(g,p,p',r),s_2) \wedge$
$Poss(end\_Grundbuchverfahren(g,s_3) \wedge s'=do(\ end\_Grundbuchverfahren(g),s_3)$
$=$
$r=Eigentum \wedge$
$Poss(start\_Grundbuchverfahren(g),s) \wedge$
$Poss(register\_Dokument(g,p',p,r),do(start\_Grundbuchverfahren(g),s)) \wedge$
$Poss(transfer\_Recht(g,p,p',r),do(register\_Dokument(g,p',p,r),do(start\_Grundbuchverfahren(g),s))) \wedge$
$Poss(end\_Grundbuchverfahren(g,do(transfer\_Recht(g,p,p',r),do(register\_Dokument(g,p',p,r),$
$do(start\_Grundbuchverfahren(g),s))))) \wedge$
$s'=do(\ end\_Grundbuchverfahren(g),do(transfer\_Recht(g,p,p',r),do(register\_Dokument(g,p',p,r),$
$do(start\_Grundbuchverfahren(g),s)))))$

Für diese Formel wird bezüglich der Updatespezifikation ein Beweis gesucht (zum Beispiel: $Axioms \vdash \exists s Do(Eigentumstransfer(G,P1,P2,Eigentum),S_0,s))$. Der Beweis liefert eine Variablenbindung für s. s ist eine Folge von Situationen, die der Ausführung einer Anzahl von bestimmten Transaktionen entspricht. Also erhält man eine Folge von Transaktionen, die zu der Eintragung des Eigentumstransfers führt, wenn eine solche Folge existiert.
Die Bindung für s, die, wenn sie existiert, gefunden wird, liefert eine Transaktionsfolge, in der jede Transaktion in der Situation möglich (poss) ist, die mit ihr assoziiert ist. Das heißt, daß die Auswertung einer GOLOG Prozedur eine ausführbare Kette von Transaktionen liefert.

4.6.3 Repräsentation von Zeit

Der Situationskalkül in der bisher präsentierten Form hat den Nachteil, daß der dort abgebildete Zeitbegriff rein qualitativ, das heißt nur durch die Ordnung der Situationen gegeben ist. Für die Repräsentation eines Grundbuchsystems ist es notwendig, die Zeit auch quantitativ anzugeben, da auch die exakten Zeitpunkte von Ereignissen von Bedeutung sind. Um dies zu ermöglichen, ist es notwendig, einige Erweiterungen an dem beschriebenen Ansatz vorzunehmen. Dies führt zum Sequentiellen Temporalen Situationskalkül (Reiter 1998b). Es soll repräsentiert werden, daß eine bestimmte Transaktion zu einem Zeitpunkt t ausgeführt wird. Hierzu erhält jede Aktion ein neues Argument, das die Zeit bezeichnet, zu der die Transaktion ausgeführt wird. Es wird das Funktionssymbol $time: Transactions \rightarrow Integers$ eingeführt mit

$$time(A(x_1,...,x_n,t))=t \tag{4.22}$$

Für jede Transaktion A wird ein solches Axiom hinzugefügt.
Ein weiteres Funktionssymbol $start: Situations \rightarrow Integers$ mit

$$start(do(a,s))=time(a) \tag{4.23}$$

bezeichnet den Beginn einer Situation.
Es ist notwendig, die Definition der Relation < zwischen Situationen zu aktualisieren (Axiome (4.4) und (4.5)). Dazu wird Axiom (4.5) wie folgt verändert:

$$\forall a,s,s' \; s<do(a,s') \Leftrightarrow Poss(a,s') \wedge s \leq s' \wedge start(s') \leq time(a) \tag{4.24}$$

s < s' bedeutet mit dieser Veränderung, daß man von s nach s' durch eine Folge von in der jeweiligen Situation möglichen Transaktionen gelangt und daß die Zeit sich dabei nicht verringert. Welche Zeit einer Situation zugeordnet wird, ist nicht bestimmt und hängt von der jeweiligen Applikation ab. Insbesondere bestimmt sie die Anfangszeit, also die der Situation S_0.
Zu beachten ist, daß durch diese Formalisierung Transaktionen möglich sind, die zwar in der Situationsfolge hintereinander kommen, aber dieselbe Transaktionszeit besitzen, also quasi gleichzeitig ausgeführt werden.

Beispiel 4.1 Ausschnitt aus einem Grundbuch (fortgesetzt)

Der Spezifikation werden die folgenden Axiome hinzugefügt:

$time(register\_Dokument(...,t))=t$
$time(transfer\_Recht(...,t))=t$
$time(start\_Grundbuchverfahren(...,t))=t$
$time(end\_Grundbuchverfahren(...,t))=t$

4.6.4 GOLOG mit externen Transaktionen und Reaktionen

Dynamische Systeme existieren nicht isoliert von der Außenwelt. So ist es auch ein wichtiger Punkt bei der Repräsentation des Grundbuchsystems, äußere Einflüsse aufzunehmen. Die Analyse des Grundbuchsystems ergab, daß äußere Einflüsse durch externe elementare

Operationen abgebildet werden können. Die Charakteristik dieser externen Transaktionen war es, daß sie zwar in ihrer Wirkung auf das System bestimmt sind, nicht aber in ihrem Auftreten. Das heißt, daß im System nicht spezifizierbar ist, wann sie auftreten. Sie können also auch nicht Bestandteil einer komplexen Transaktion sein, denn das würde bedeuten, daß sie explizit aufgerufen würden. Wie kann nun formal beschrieben werden, daß externe Transaktionen eintreten und das System verändern?
Hierzu wird GOLOG zu RGOLOG[4] (Reiter 1998a, S.203ff) erweitert. Der Grundgedanke ist folgender: Man stellt sich vor, es würden zwei Programme laufen. Eins beschreibt die internen Operationen des Systems, das andere die Operationen außerhalb des Systems. Sie laufen gleichzeitig ab, das heißt, es können Transaktionen des internen und des externen Programms geschehen. GOLOG muß nun so erweitert werden, daß sowohl interne als auch externe Transaktionen stattfinden können. Dies kann dadurch erreicht werden, daß nach jeder Ausführung einer internen Transaktion auch die Ausführung einer externen Transaktion möglich ist. Auf diese Art und Weise werden das externe und das interne Programm miteinander verschachtelt. Es ist auch so vorstellbar, daß nach Ausführung einer Anweisung das interne Programm unterbrochen und das externe Programm ausgeführt wird. Danach wird das interne Programm weiter ausgeführt.
Dieser Gedanke soll nun durch Veränderung der Definition von GOLOG umgesetzt werden. Der Ausgangspunkt ist die Frage, wie die Semantik zweier zeitlich verschachtelter GOLOG Programme aussieht. Dazu wird das Makro $Do1(\alpha,Q,s,s')$ induktiv über die Struktur des ersten Arguments definiert. Hierbei ist α ein beliebiges GOLOG Programm und Q ein binäres Prädikat über Situationen.
Wenn α eine elementare Transaktion ist, dann gilt:

$$Do1(\alpha,Q,s,s') \stackrel{def}{=} Poss(\alpha[s],s) \wedge Q(do(a,s),s') \tag{4.25}$$

Ist α keine elementare Transaktion, so gilt

$$Do1(?\varphi,Q,s,s') \stackrel{def}{=} \varphi[s] \wedge s=s'$$

$$Do1([\alpha_1;\alpha_2],Q,s,s') \stackrel{def}{=} \exists s^* \; Do1(\alpha_1,Q,s,s^*) \wedge Do1(\alpha_2,Q,s^*,s')$$

$$Do1(\alpha_1 \mid \alpha_2,Q,s,s') \stackrel{def}{=} Do1(\alpha_1,Q,s,s') \vee Do1(\alpha_2,Q,s,s')$$

$$Do1((\pi x)\alpha(x),Q,s,s') \stackrel{def}{=} \exists x \; Do1(\alpha(x),Q,s,s')$$

$$Do1(\alpha^*,Q,s,s') \stackrel{def}{=} \forall P \; (\forall s_1 \; (P(s_1,s_1)) \wedge \forall s_1 \forall s_2 \forall s_3 \; (P(s_1,s_2) \wedge Do(\alpha,Q,s_2,s_3) \Rightarrow P(s_1,s_3)))$$

$$\underline{if}\ \varphi\ \underline{then}\ \alpha_1\ \underline{else}\ \alpha_2\ \underline{endif} \stackrel{def}{=} (?\varphi;\alpha_1) \mid (?\neg\varphi;\alpha_2)$$

$$\underline{while}\ \varphi\ \underline{do}\ \alpha\ \underline{endwhile} \stackrel{def}{=} (?\varphi;\alpha)^*;?\neg\varphi$$

Prozeduren:

$$Do1(P(t_1,...,t_n),Q,s,s') \stackrel{def}{=} P(t_1[s],...,t_n[s],Q,s,s')$$

Programme:

$$\underline{proc}\ P_1(v_1)\alpha_1\ \underline{endProc}\ ;...;\ \underline{proc}\ P_n(v_n)\alpha_n\ \underline{endProc}\ ;\ \alpha_0$$

[4] Reactive GOLOG

$$Do1(\{\underline{proc}\ P_1(v_1)\alpha_1\ \underline{endProc}\ ;...;\ \underline{proc}\ P_n(v_n)\alpha_n\ \underline{endProc}\ ;\ \alpha_0\},Q,s,s') \stackrel{def}{=}$$

$$\forall P_1,...,P_n(\bigwedge_{i=1}^{n}(\forall s_1,s_2,v_i\ (Do1(\alpha_i,Q,s_1,s_2)\Rightarrow P_i(v_i,s_1,s_2)))\Rightarrow Do1(\alpha_0,Q,s,s'))$$

Dies unterscheidet sich von der Definition von GOLOG nur im Fall, daß α eine elementare Transaktion ist. *Do(a,s,s')* beschreibt einen Übergang von Situation *s* in Situation *s'=do(a,s)*. Im Unterschied dazu beschreibt *Do1(a,Q,s,s')* zuerst einen Übergang von *s* in die Situation *do(a,s)* und danach in eine Situation s', in der das Prädikat *Q(do(a,s),s')* gilt. Intuitiv drückt *Do1* also die Tatsache aus, daß nach Ausführung jeder elementaren Transaktion *a* ein weiterer Situationsübergang geschieht. Dieser weitere Situationsübergang ist verursacht durch das parallel laufende Programm. Nun muß dieses parallel laufende Programm und seine Beziehung zu dem Prädikat *Q* spezifiziert werden. Dazu wird das parallel laufende Programm mit *rules* bezeichnet und das Makro *DoR(A,rules,s,s')* wie folgt definiert:

$$DoR(A,rules,s,s') \stackrel{def}{=} \forall Q(\forall s_1,s_2(Do1(rules,Q,s_1,s_2)\Rightarrow Q(s_1,s_2))\Rightarrow Do1(A,Q,s,s')) \quad (4.26)$$

Q ist nun der Situationsübergang, der durch Evaluierung von *rules* entsteht. So bezeichnet *DoR(A,rules,s,s')* den Situationsübergang, der dadurch verursacht wird, daß nach jeder elementaren Transaktion aus *A* ein Situationsübergang durch Evaluierung von *rules* erfolgt. Also beschreibt *DoR* den Sachverhalt, daß nach jeder elementaren Transaktion aus *A* das Programm *rules* ausgeführt wird.
Was für eine Rolle spielt das Programm *rules* in dem Fall, daß externe Transaktionen aufgenommen werden sollen? Es müßte eine Möglichkeit enthalten, externe Transaktionen einzugeben. Die Art und Weise, wie diese Eingabe erfolgt, ist eine Frage der konkreten Umsetzung, also der Implementation. Ergebnis dieser Eingabe muß eine externe Transaktion sein, die dann parallel zu dem internen Programm ausgeführt wird. Beispielsweise könnte *rules* wie folgt aussehen:

```
proc rules()
    eingabe_externe_Transaktion ()
endProc
```

Neben der Aufnahme von externen Transaktionen muß das System in der Lage sein, auf die sich damit verändernden Bedingungen automatisch zu reagieren. Zum Beispiel muß ein auf Lebenszeit begrenztes Recht automatisch aus dem Grundbuch gelöscht werden, wenn der Tod der Person bekannt wird. Das System muß also in der Lage sein, solche Bedingungen zu überwachen und dann Transaktionen auszuführen. Das geeignete Konzept, dies umzusetzen, heißt Interrupts. Tritt eine bestimmte Bedingung während des Programmlaufs ein, stoppt das Programm und reagiert auf die eingetretene Bedingung. Dieser Gedanke wird nun dadurch umgesetzt, daß die Unterbrechungssteuerung in Form von Regeln formuliert wird. Zum Beispiel der Form:

Eingetragenes_Recht(grundstück,person,recht,s)$\wedge\neg$*Person(person,s)*$\Rightarrow$
lösche_eingetragenes_recht(grundstück,person,recht)

Wie kann nun eine Menge von Unterbrechungen in einer GOLOG Prozedur formuliert werden?
Dazu wird die Prozedur *interrupts()* wie folgt eingeführt:

```
proc interrupts()
    ?Φ1;a1 |
    ...
    ?Φn;an |
    ?¬(Φ1∨...∨Φn)
endProc
```

Dabei bezeichnen $?\Phi_i;a_i$ die Regeln für die Unterbrechungen. Die letzte Regel $?\neg(\Phi_1 \vee ... \vee \Phi_n)$ sorgt dafür, daß ein Programmlauf möglich ist, auch wenn keine der Regeln zutreffend sind. Für das obige Beispiel sieht die Prozedur *interrupts* wie folgt aus:

```
proc interrupts()
    ?(Eingetragenes_Recht(g,p,r)∧¬Person(p));
    lösche_eingetragenes_recht(grundstück,person,recht) |
    ?(¬(Eingetragenes_Recht(g,p,r)∧¬Person(p)))
endProc
```

Abschließend muß die Prozedur aufgerufen werden. Dazu wird die Prozedur *rules* ergänzt:

```
proc rules()
    interrupts();
    eingabe_externe_Transaktion ()
endProc
```

4.7 Anfragen an das System

Wie kann eine Antwort auf eine Anfrage an des System in einer Situation s gefunden werden? Die Datenbank befindet sich im Zustand s, wenn bezüglich der initialen Datenbank eine Folge von Transaktionen ausgeführt wird, die im Zustand s endet. Die Anfrage kann also nur bezüglich der Liste der ausgeführten Transaktionen beantwortet werden. Um die Antwort zu finden, muß dann bestimmt werden, ob die Formel, die die Anfrage formuliert, gültig ist in dem Zustand s. Gegeben ist eine Anfrage (Query) Q(s), deren einzige freie Variable s der Sorte *Situations* und $t_1,...,t_n$ eine Folge von Grundtermen der Sorte *Transactions* ist. Die Antwort auf die Query wird nun dadurch bestimmt, daß die folgende Formel bewiesen wird:

$$D \models Q(do([t_1,...,t_n],S_0))$$

Theorem 4.1 (Reiter 1995, Theorem 5.2)
Sei Q(s) eine einfache Formel mit der einzigen freien Variablen der Sorte Situations, und $t_1,...,t_n$ eine Folge von Grundtermen der Sorte Transactions. Wenn $t_1,...,t_n$ eine bezüglich der Datenbank D gültige Folge von Transaktionen ist, dann gilt

$$D \models Q(do([t_1,...,t_n],S_0)) \Leftrightarrow D_{una} \cup D_{S0} \models R[Q(do([t_1,...,t_n],S_0))]$$

Auch hier reduziert sich die Query Evaluierung auf einen Beweis bezüglich der initialen Datenbank.

4.8 Diskussion und Zusammenfassung

Am Anfang dieses Kapitels stand die Überlegung, daß die Repräsentation von Wissen über ein System als Spezifikation dieses Systems betrachtet werden kann. Speziell kann die Repräsentation von Wissen auf Basis des Situationskalküls als Spezifikation von Datenbanken und der sie verändernden Transaktionen angesehen werden. In diesem Kapitel wurde ausgegangen von den Anforderungen, die die Modellierung eines Grundbuchsystems an die Repräsentationssprache stellt. Daraufhin wurden die formalen Grundlagen eingeführt, die notwendig sind, um diesen Anforderungen gerecht zu werden. Es stellt sich die Frage, welche allgemeinen Probleme des Situationskalküls damit gelöst wurden und ob diese für die Spezifikation einer Datenbank überhaupt relevant sind. Dies soll im folgenden diskutiert werden.

4.8.1 Warum Logik zur Spezifikation von Datenbanken?

Aus der Sicht der Wissensrepräsentation ist bereits motiviert worden, warum die Prädikatenlogik eine geeignete Sprache zur Formulierung von Wissen über die Welt oder ein

Problem ist. In Bezug auf die Spezifikation von Datenbanken lassen sich die Vorteile in den folgenden Punkten präzisieren (Reiter 1984):

1. Logik ist genau und eindeutig. Sie hat eine genau definierte Syntax und Semantik und verfügt deshalb über die wichtige Verbindung von Formeln und ihrer Bedeutung, also der Beschreibung der Welt und der Welt selbst.
2. Logische Datenmodelle bieten ein hohe Stufe der Abstraktion. Sie sind nicht prozedural und beschränken sich darauf, die Eigenschaften des Modells anzugeben, nicht wie es umgesetzt werden kann.
3. Das Datenmodell ist transparent. Das Wissen, das repräsentiert wird, kann inspiziert und überprüft werden.
4. Da logische Datenmodelle Spezifikationen sind, können sie in vielfältiger Weise realisiert werden in einem prozedural orientierten Datenmodell. Diese Umsetzung kann bezüglich ihrer Korrektheit und Vollständigkeit bewiesen werden.

4.8.2 Warum ist eine Lösung des Frame Problems notwendig?

Die Repräsentation der Invarianten eines sich verändernden Gegenstandsbereichs ist in der Wissensrepräsentation ein zentraler Forschungsgegenstand. Warum ist diese Frage in dem Fall der Spezifikation von Datenbanken im Situationskalkül relevant? McCarthy hat erkannt, daß das Frame Problem im Situationskalkül unter Verwendung sogenannter Frame Axiome gelöst werden kann (McCarthy und Hayes 1969). Für jedes Fluent-Aktions-Paar ist ein solches erforderlich, das angibt, daß die Aktion das Fluent nicht verändert. Aus Sicht der Komplexität ist dies problematisch, wenn viele Transaktionen und viele Fluents vorhanden sind, aber die Transaktionen jeweils nur einen kleinen Teil der Fluents beeinflussen. Aus diesem Grund wurde der Situationskalkül für ungeeignet zur Lösung praktischer Probleme angesehen. Die Reitersche Updatespezifikation löst dieses Problem insofern, als daß diese Frame Axiome nicht explizit angegeben werden müssen, sondern implizit in den Successor State Axiomen enthalten sind. Folglich sind viel weniger Axiome bei der Repräsentation notwendig, was die Komplexität senkt.

Ist nun dieses Argument auch aus der Spezifikationssicht relevant? Im besonderen Fall: Ist es für die Spezifikation eines Grundbuchsystems relevant? Entsprechend dem im Kapitel 3 entwickelten konzeptionellen Modell sicher nicht: Es gibt nur eine kleine Anzahl von Transaktionen und Fluents. In (Borgia, Mylopoulos et al. 1995) ist beschrieben, daß im Falle der Spezifikation von Programmen durch die Angabe von Vor- und Nachbedingungen sehr wohl die Beschreibung lang und unübersichtlich werden kann, was den Anforderungen an eine Spezifikationssprache klar zuwider läuft. Der hier verwendete Situationskalkül ist genau solch ein Ansatz, der Vor- und Nachbedingungen zur Spezifikation des Systems nutzt. Die Lösung des Frame Problems, das zu der beschriebenen Updatespezifikation führt, hat nun folgende Vorteile (Reiter 1998a, S.32):

1. Modularität. Kommen neue Transaktionen oder Fluents hinzu, müssen nur die Effektaxiome angegeben werden. Die Updatespezifikation kann automatisch gewonnen werden und berücksichtigt alle Veränderungen.
2. Exaktheit. Es können keine Axiome, insbesondere Frame Axiome, vergessen werden.

Man kann sagen, daß die Lösung des Frame Problems die Grundlage schafft für die Spezifikation von Datenbank Updates. Erst diese Lösung macht den Situationskalkül geeignet zur Nutzung als Spezifikationssprache.

4.8.3 Angabe der Effektaxiome vs. Angabe der Updatespezifikation

Zwei Möglichkeiten bestehen bei der Spezifikation. Es können die Effektaxiome angegeben werden, woraus sich dann im vorgestellten Verfahren die Spezifikation ergibt oder man kann die Updatespezifikation direkt formulieren. Für die erste Variante sprechen genau die Argumente, die im vorigen Abschnitt angegeben wurden: Modularität und Exaktheit. Man kann sie aber auch auf die zweite Möglichkeit übertragen. Die Updatespezifikation selbst

bietet ebenfalls diese Vorteile. Sie enthält für jedes Fluent genau ein Axiom, das Veränderungen dieses Fluents beschreibt. Für jede Transaktion besteht genau ein Axiom, das die Vorbedingungen für diese Transaktion angibt. Transaktionen und Fluents können einfach hinzugefügt werden, indem jeweils die zugehörenden Axiome formuliert werden. Die einzige Schwierigkeit ist, daß neue Transaktionen auch Wirkungen auf bestehende Fluents haben können und folglich deren Spezifikation verändert werden muß. Da sich derjenige, der die Spezifikation erstellt, sehr klar über die zu beschreibenden Transaktionen sein muß, ist dies zu vertreten. Das Argument aus dem vorigen Abschnitt bleibt bestehen: Die Lösung des Frame Problems ermöglicht die Erstellung übersichtlicher Spezifikationen. In dieser Arbeit wird auf Grund der vorgetragenen Argumentation der zweite Weg bevorzugt, die Updatespezifikation wird direkt angegeben.

4.8.4 Das Beweisen von Eigenschaften der Spezifikation

Die Eigenschaften einer Updatespezifikation lassen sich mit den Mitteln der Prädikatenlogik beweisen. Was gilt nun aber für GOLOG Programme? In dieser Arbeit werden sie zur Spezifikation von komplexen Transaktionen genutzt. Eigenschaften von GOLOG Programmen lassen sich ebenfalls mit den Mitteln der Prädikatenlogik beweisen, da sie eine klare Semantik im Situationskalkül haben. Ein GOLOG Programm ist eine Abkürzung für eine Formel im Situationskalkül. Bei Evaluierung eines solchen Programms erfolgt die Übersetzung in den Situationskalkül. Für diese Formel gilt dann die normale Semantik des Situationskalküls. Eigenschaften, die sich bezüglich eines Programms beweisen lassen, sind (partielle) Korrektheit und Terminierung. Ein Hauptvorteil der hier verwendeten Methode ist der folgende: Elementare und komplexe Transaktionen haben dieselbe formale Grundlage. So kann Terminierung eines GOLOG Programms auf die Frage zurückgeführt werden, ob sich die ihm zugrunde liegende Formel beweisen läßt. Korrektheit reduziert sich auf die Frage, ob, wenn ein Programm terminiert, es dies in einer Situation tut, die bestimmte Eigenschaften hat.

4.8.5 Gelöste und ungelöste Probleme des Situationskalküls

Der vorgestellte Ansatz löst eine Reihe von Problemen, die zunächst gegen seine Verwendung sprachen: das Frame Problem, Abbildung von Zeit, Zeitdauer und Gleichzeitigkeit von Aktionen. Soweit es für die Repräsentation des Grundbuchs relevant war, wurden die Lösungen präsentiert. Ein in der Künstlichen Intelligenz oft betrachtetes und bereits erwähntes Problem ist das *Ramification* Problem. Hierunter wird das Problem verstanden, die indirekten Effekte von Aktionen zu bestimmen. Es wurde hier nicht behandelt, obwohl es im Situationskalkül dazu Lösungen gibt (McIlraith 1997). Das hat folgenden Grund: Es ist vom Standpunkt des Grundbuchrechts sinnvoll, die Wirkungen von Veränderungen explizit festzulegen, um Rechtssicherheit zu schaffen. Implizite, indirekte Effekte sind nicht erwünscht. Dies ist natürlich in der Rechtsprechung umgesetzt worden. Zur Repräsentation des Grundbuchsystems ist es deshalb hinreichend, sich auf direkte Wirkungen von Aktionen zu beschränken.

5 Das formale Modell

5.1 Einleitung

Auf Basis der formalen Grundlagen ist es nun möglich, das konzeptionelle Modell formal zu beschreiben. Ausgangspunkt für die formale Beschreibung ist der sequentielle temporale Situationskalkül, wie er in Kapitel 4 beschrieben wurde. Die mehrsortige Logik bestand aus Sorten für Situationen, Transaktionen und Objekten. In diesem Abschnitt wird die Sorte Objekte weiter unterteilt entsprechend den Anforderungen des Modells. Zur Modellierung von Zeit werden zum Beispiel Integer genutzt. Werte eines Rechts werden als Real-Werte modelliert. Das konzeptionelle Modell hat drei Bestandteile: das Modell der materiellen Rechtsverhältnisse, das Modell des Grundbuchs und das Modell der Beziehungen zwischen beiden. Dieses Kapitel folgt dieser Einteilung. Das Modell der materiellen Rechtsverhältnisse wird auf Basis elementarer Transaktionen beschrieben. Um aufzuzeigen, daß es sich bei Veränderungen in diesem Modell um Beobachtungen des Handelns von Personen in der realen Welt handelt, werden diese Veränderungen durch Aktionen beschrieben. Aktionen bilden das Agieren von Personen in der Welt ab. Die Unterscheidung von Transaktionen des Grundbuchs ist rein konzeptioneller Natur, die Formalisierung erfolgt auf derselben Grundlage. Nach der Beschreibung der materiellen Rechtsverhältnisse erfolgt die Beschreibung des Grundbuchs. Hierzu werden die elementaren und komplexen Transaktionen entsprechend dem konzeptionellen Modell formuliert. Die Beziehungen dieser beiden Modelle werden in Hinblick auf die Frage der Grundbuchunrichtigkeit betrachtet. Formal beschrieben wird diese Beziehung durch Inkonsistenzmarkierungen, die die Nichtübereinstimmung der beiden Modelle anzeigen. Anschließend wird an einigen Beispielen motiviert, welche Probleme, die sich aus der Grundbuchunrichtigkeit ergeben, durch die Formalisierung beschrieben werden können. Abschluß des Kapitels bildet die Beschreibung von Transaktionen zur Korrektur von unrichtigen Stellen im Grundbuch auf Basis der Inkonsistenzmarkierungen.

5.2 Das Modell der materiellen Rechtsverhältnisse

In diesem Abschnitt werden die materiellrechtlichen Verhältnisse formal beschrieben, die Grundlage für die Abbildung im Grundbuch sind. Zentral ist die Frage, wie sich diese Rechtsverhältnisse verändern. Hierzu werden die Objekte, sowie die Beziehungen zwischen Objekten und Aktionen, die die Beziehungen verändern, beschrieben. Objekte und Beziehungen werden als Relationen modelliert und in der Sprache des Situationskalküls als Fluents abgebildet. Die Formalisierung von Aktionen und ihren Wirkungen auf die Fluents erfolgt in einer Updatespezifikation, wie sie in Kapitel 4 beschrieben wurde.

5.2.1 Objekte

Objekte des materiellen Rechtsverkehrs sind Grundstücke, Personen und Flurstücke. Entsprechend der Argumentation aus 3.5.1 ist es vom zeitlichen Kontext abhängig, ob ein Bezeichner einem Objekt der materiellen Welt zugeordnet ist. Dem wird durch die Modellierung der Objekte als Fluents Rechnung getragen. Fluents, die Objekte sind, bilden Relationen zwischen Situationen und Bezeichnern ab. Sie stellen damit die Beziehung zwischen Bezeichnern im Grundbuch und Objekten der materiellen Welt her.
Auf die Formalisierung bezogen muß bemerkt werden, daß Objekte in diesem Sinn nicht mit Objekten der Sprache des Situationskalküls gleichgesetzt werden können.

> $Grundstück \subseteq Grundstück\_ID \times Situations$
> *Grundstück(g,s)* bedeutet, daß g in der Situation s ein Grundstück bezeichnet.

Person ⊆ *Person_ID* x *Situations*
Person(p,s) bedeutet, daß durch p in der Situation s eine Person bezeichnet ist.
Flurstück ⊆ *Flurstück_ID* x *Situations*
Flurstück(f,s) bedeutet, daß durch f in der Situation s ein Flurstück bezeichnet ist.

Zur Einteilung der Objekte der Sprache (zum Beispiel *ID_Grundstück*) wird auf die Beschreibung des Modells des Grundbuchs im Abschnitt 5.3 verwiesen. Wichtig ist zu diesem Zeitpunkt, daß die Verbindung von Bezeichnern von Objekten und Objekten des materiellen Rechtsverkehrs hergestellt wird. Zum Beispiel beschreibt *Grundstück(g,s)* die Tatsache, daß der Bezeichner g in der Situation s einem Grundstück zugeordnet ist.

5.2.2 Relationen

Relationen in dem hier gebrauchten Sinn bezeichnen die Beziehungen zwischen Objekten des materiellen Rechtsverkehrs. Sie werden als Fluents modelliert. Damit wird der Tatsache Rechnung getragen, daß Rechtsverhältnisse zeitlichen Veränderungen unterliegen.

Eigentümer(g,p,s) ist eine Relation zwischen einem Grundstück und einer Person. Es bezeichnet die Tatsache, daß die Person p in der Situation s Eigentümer des Grundstücks g ist.
Gläubiger_Hypothek(g,p,w,s) ist eine Relation zwischen einem Grundstück, einer Person und einem Wert. Es bezeichnet die Tatsache, daß die Person p in der Situation s Gläubiger einer Hypothek mit den Wert w am Grundstück g ist.
Nießbraucher(g,p,s) ist eine Relation zwischen einem Grundstück und einer Person. Es bezeichnet die Tatsache, daß die Person p in der Situation s das Nutzungsrecht am Grundstück g hat.
Grundstück_Flurstücke_m(g,fl,s) ist eine Relation zwischen einem Grundstück und einem Flurstück. Dadurch wird bezeichnet, daß das Flurstück f zum Grundstück g gehört. *"_m"* zeigt an, daß es sich um die Relation zwischen Grundstücken und Flurstücken in der materiellen Welt handelt in Abgrenzung zur analogen Relation im Grundbuch.

5.2.3 Aktionen

Aktionen bezeichnen Veränderungen in den materiellen Rechtsverhältnissen. Sie stellen Beobachtungen von Tatsachen dar, also Beobachtungen von Veränderungen, die in der Welt stattfinden und im Grundbuch abgebildet werden müssen. Aktionen beschreiben das Handeln von Personen in der Welt, welches die materiellen Rechtsverhältnisse verändert.

verkaufe_Grundstück: Grundstück_ID x *Person_ID* x *Person_ID* x *Integers* → *Transactions*
verkaufe_Grundstück(g,p1,p2,t) bezeichnet den Vorgang, daß das Grundstück g von der Person p1 an die Person p2 zum Zeitpunkt t verkauft wird.
bestelle_Hypothek: Grundstück_ID x *Person_ID* x *Person_ID* x *Reals* x *Integers* → *Transactions*
bestelle_Hypothek(g,p1,p2,w,t) bezeichnet den Vorgang, daß eine Person p1 zu Lasten der Person p2 eine Hypothek am Grundstück g bestellt.
zurückzahlen_Hypothek: Grundstück_ID x *Person_ID* x *Person_ID* x *Reals* x *Integers* → *Transactions*
zurückzahlen_Hypothek(g,p1,p2,w,t) bezeichnet den Vorgang, daß eine Person p1 die auf Grundstück g lastende Hypothek mit dem Wert w an die Person p2 zum Zeitpunkt t zurückzahlt.

bestelle_Nießbrauch: Grundstück_ID x *Person_ID* x *Person_ID* x *Integers* → *Transactions*
bestelle_Nießbrauch(g,p1,p2,t) bezeichnet den Vorgang, daß Person p1 der Person p2 zum Zeitpunkt t das Nutzungsrecht an seinem Grundstück g überläßt.
vereinige_Grundstück: Person_ID x *Grundstück_ID* x *Grundstück_ID* x *Integers* → *Transactions*
vereinige_Grundstück(p,g1,g2,t) bezeichnet den Vorgang, daß Person p sein Grundstück g1 mit dem Grundstück g2 zum Zeitpunkt t vereinigt. Das Grundstück g2 hört dabei auf zu bestehen.
teile_Grundstück: Person_ID x *Grundstück_ID* x *Grundstück_ID* x *Flurstück_ID* x *Integers* → *Transactions*
teile_Grundstück(p,g1,g2,f,t) bezeichnet den Vorgang, daß Person p von seinem Grundstück g1 das Flurstück f zum Zeitpunkt t abteilt, das dann als Grundstück g2 geführt wird.
stirbt_Person: Person_ID x *Integers* → *Transactions*
stirbt_Person (p,t) bezeichnet den Vorgang, daß die Person p zum Zeitpunkt t stirbt.

5.2.3.1 Vorbedingungen für Aktionen: Transaction Precondition Axiome

Aktionen sollen Beobachtungen über Handlungen von Personen in der materiellen Welt sein. Es handelt sich dabei um Handlungen, die im Modell der materiellen Rechtsverhältnisse abgebildet werden. Diese Handlungen sind demzufolge real geschehen. Vorbedingungen müssen also zwei Anforderungen erfüllen: Sie müssen einerseits nur Handlungen zwischen existierenden Objekten zulassen und müssen andererseits die Einhaltung der materiellrechtlichen Voraussetzungen sicherstellen. So kann zum Beispiel nur eine lebende Person ein Grundstück verkaufen und auch nur, wenn sie Eigentümer dieses Grundstücks ist.

Poss(verkaufe_Grundstück(g,p1,p2,t),s)⇔
Grundstück(g,s)∧Person(p1,s)∧Person(p2,s)∧Eigentümer(g,p1,s)
Nur der Eigentümer eines Grundstücks kann ein Grundstück verkaufen.
Poss(bestelle_Hypothek(g,p1,p2,w,t),s)⇔
Grundstück(g,s)∧Person(p1,s)∧Person(p2,s)∧w>0∧Eigentümer(g,p2,s)
Eine Hypothek kann nur zu Lasten des Eigentümers eines Grundstücks bestellt werden. Ihr Wert muß positiv sein.
Poss(zurückzahlen_Hypothek(g,p1,p2,w,t),s)⇔Grundstück(g,s)∧Person(p1,s)∧ Person(p2,s)∧Eigentümer(g,p1,s)∧Gläubiger_Hypothek(g,p2,w,s)
Eine Hypothek kann nur vom Eigentümer eines Grundstücks an den Gläubiger zurückgezahlt werden.
Poss(bestelle_Nießbrauch(g,p1,p2,t),s)⇔
Grundstück(g,s)∧Person(p1,s)∧Person(p2,s)∧Eigentümer(g,p1,s)
Ein Nießbrauch kann nur vom Eigentümer eines Grundstücks zu Gunsten einer anderen Person bestellt werden.
Poss(vereinige_Grundstück(g1,g2,p,t),s)⇔
Grundstück(g1,s)∧Grundstück(g2,s)∧Person(p,s)∧Eigentümer(g1,p,s)∧ Eigentümer(g2,p,s)
Die Vereinigung im eigenen Besitz kann vorgenommen werden.
Poss(teile_Grundstück(g1,g2,p,f,t),s)⇔
Grundstück(g1,s)∧Person(p,s)∧Flurstück(f,s)∧¬Grundstück(g2,s)∧
Eigentümer(g1,p,s)∧ Grundstück_Flurstücke_m(g1,f,s)∧ ∃f2
Grundstück_Flurstücke_m(g1,f2,s)∧f≠f2

Die Teilung im eigenen Besitz kann erfolgen, wenn durch die Abteilung eines Flurstücks nicht das Ausgangsgrundstück aufhört zu bestehen. Das heißt, daß das Ausgangsgrundstück nach der Teilung noch Flurstücke in seinem Bestand haben muß.

Poss(stirbt_Person (p,t),s)⇔ Person(p,s)

Nur eine lebende Person kann sterben. Entsrechendes gilt natürlich auch allgemein für juristische Personen.

5.2.3.2 Wirkungen von Aktionen: Successor State Axiome

Wirkungen von Aktionen bezeichnen entweder Veränderungen, die die Objekte selbst oder die materiellen Rechtsverhältnisse zwischen diesen Objekten betreffen.

- Wirkungen, die Objekte betreffen:

  Poss(a,s)⇒(Person(p,do(a,s))⇔Person(p,s)∧¬a=stirbt_Person(g,t))

  Eine Person hört mit dem Tod auf zu existieren.

  Bemerkung: In diesem Modell sollen keine Personen hinzukommen.

  Poss(a,s)⇒(Grundstück(g,do(a,s))⇔ a=teile_Grundstück(g',g,p',f,t)∨
  Grundstück(g,s)∧ ¬a=vereinige_Grundstück(g',g,p'',t))

  Ein Grundstück entsteht, wenn ein Teil von einem anderen Grundstück abgeteilt und daraus ein neues Grundstück gebildet wird. Ein Grundstück hört auf zu bestehen, wenn es durch Vereinigung einem anderen Grundstück zugeordnet wird.

  Poss(a,s)⇒(Flurstück(f,do(a,s))⇔Flurstück(f,s))

  Alle Flurstücke, bleiben im Modell erhalten.

  Bemerkung: Flurstücksveränderungen geschehen im Kataster, dies ist nicht Gegenstand dieser Arbeit.

- Wirkungen, die die materiellen Rechtsverhältnisse betreffen:

  Poss(a,s)⇒(Eigentümer(g,p,do(a,s))⇔
  a=verkaufe_Grundstück(g,p',p,t)∨a=teile_Grundstück(g',g,p''f,t)∨
  Eigentümer(g,p,s)∧¬a=verkaufe_Grundstück(g,p,p'',t)∧
  ¬a=vereinige_Grundstück(g',g,p''',t))

  Die Person p wird Eigentümer des Grundstücks g, wenn er es von p' kauft, es durch Teilung im eigenen Besitz entsteht, oder p es bereits besitzt und nicht weiterverkauft hat und es nicht durch Vereinigung mit einem anderen Grundstück aufhört zu bestehen.

  Poss(a,s)⇒(Gläubiger_Hypothek(g,p,w,do(a,s))⇔
  a=bestelle_Hypothek(g,p,p',w,t)∨a=vereinige_Grundstück(g,g',p''',t)∧
  Gläubiger_Hypothek(g',p''',w,s)∨
  Gläubiger_Hypothek(g,p,w,s)∧¬a=zurückzahlen_Hypothek(g,p'',p,t)∧
  ¬a=vereinige_Grundstück(g'',g,p'''',t))

  Eine Person p wird Gläubiger einer Hypothek an einem Grundstück g, wenn er sie zu Gunsten einer Person p' bestellt, die Hypothek durch Vereinigung auf dieses Grundstück übertragen wird, oder er bereits Gläubiger ist und die Hypothek nicht zurückgezahlt worden ist und das Grundstück nicht durch Vereinigung aufhört zu bestehen.

  Poss(a,s)⇒(Nießbraucher(g,p,do(a,s))⇔
  a=bestelle_Nießbrauch(g,p',p,t)∨a=vereinige_Grundstück(g,g',p''',t)∧
  Nießbraucher(g',p''',s)∨
  Nießbraucher(g,p,s)∧¬a=stirbt_Person(p,t)∧
  ¬a=vereinige_Grundstück(g'',g,p'''',t))

Eine Person wird Nießbraucher, wenn zu ihren Gunsten ein Nießbrauch am Grundstück bestellt wird oder der Nießbrauch durch Vereinigung auf das Grundstück übertragen wird.
Eine Person bleibt Nießbraucher dieses Grundstücks, solange sie nicht gestorben ist oder das Grundstück nicht durch Vereinigung mit einem anderen Grundstück aufhört zu bestehen.

Poss(a,s)⇒(Grundstück_Flurstücke_m(g,f,do(a,s)) ⇔
a=vereinige_Grundstück(g,g',p',t) ∧ Grundstück_Flurstücke_m(g',f,s)
∨ Grundstück_Flurstücke_m(g,f,s) ∧¬a=teile_Grundstück(g,g'',p''f,t)
∧¬a=vereinige_Grundstück(g'',g,p''',t))

Ein Flurstück ist Bestandteil des Grundstücks g, wenn es g durch Vereinigung zugeordnet wird oder bereits Bestandteil dieses Grundstücks ist und nicht von ihm abgeteilt wird und das Grundstück nicht durch Vereinigung mit einem anderen Grundstück aufhört zu bestehen.

5.2.3.3 Die initiale Datenbank

Die initiale Datenbank enthält die Definition von *time* für alle Aktionen. Weiterhin enthält sie alle Angaben über den Anfangszustand, also Angaben über die Rechtsverhältnisse bevor eine Aktion stattfand. Diese Angaben werden an dieser Stelle nicht erfolgen, da sie Teil einer konkreten Umsetzung sind.

time(verkaufe_Grundstück(g,p1,p2,t))=t
time(bestelle_Hypothek(g,p1,p2,w,t))=t
time(zurückzahlen_Hypothek(g,p1,p2,w,t))=t
time(Bestelle_Nießbrauch(g,p1,p2,t))=t
time(vereinige_Grundstück(g1,g2,p,t))=t
time(teile_Grundstück(g1,g2,p,f,t))=t
time(stirbt_Person (p,t))=t

5.3 Das Modell des Grundbuchs

Aufbauend auf dem Modell der materiellen Rechtsverhältnisse wird nun das Modell des Grundbuchs erstellt. Das Grundbuch bildet die Veränderungen in den materiellen Rechtsverhältnissen ab. Es muß also zu den Aktionen in der materiellen Welt duale Operationen im Grundbuchsystem geben. Wenn sich in der materiellen Welt Rechtsverhältnisse ändern, dann muß dies aus dem Grundbuch ersichtlich sein. Entsprechend diesen Änderungen müssen sich Eintragungen im Grundbuch ändern. Das Verhältnis von materiellem Recht und seiner Abbildung im Grundbuch bestimmt das formelle Recht. Es legt fest, wie materielles Recht und Grundbuch in Übereinstimmung gehalten werden. Grundsätzlich geschieht dies nicht automatisch, sondern auf Antrag. Als Nachweis für materielle Rechtsänderungen dienen beglaubigte Dokumente. Die Anträge und Dokumente bilden folglich die Schnittstelle zwischen materiellem Recht und Grundbuch. Aus der Sicht des Grundbuchs setzen sie das formelle Grundbuchrecht um und bilden die Schnittstelle nach außen zur materiellen Welt. Sie wurden deshalb externe Operationen genannt. Externe Operationen schaffen die Voraussetzungen für interne Operationen, die das materielle Recht abbilden. Das Zusammenspiel von formellem und materiellem Recht wird im Grundbuchverfahren festgelegt. Grundbuchverfahren schaffen die Verbindung von internen und externen Operationen. In diesem Abschnitt werden nun interne und externe Operationen beschrieben sowie ihr Zusammenwirken im Grundbuchverfahren. Hierbei sind interne und externe Operationen elementare Operationen und werden folglich als elementare Transaktionen abgebildet. Grundbuchverfahren sind komplexe Abläufe, die als komplexe Transaktionen beschrieben werden.

5.3.1 Objekte

- Rechte:

Die Sprache des Situationskalküls wird um die Sorte *Rechte* erweitert. Objekte der Sorte *Rechte* bezeichnen die im Grundbuch eintragsfähigen Rechte. Hiermit soll deutlich gemacht werden, daß nur wenige, genau festgelegte Rechte auch im Grundbuch eingetragen werden können. In diesem Modell sind die eintragsfähigen Rechte im Einzelnen das *Eigentum*, die *Hypothek* und der *Nießbrauch*.

- Grundstücke: eine Sorte *Grundstück_ID* wird eingeführt für Bezeichner von Grundstücken.
- Personen: eine Sorte *Person_ID* wird eingeführt für Bezeichner von Personen.
- Flurstücke: eine Sorte *Flurstück_ID* wird eingeführt für Bezeichner von Flurstücken.

Hiermit wird der Tatsache Rechnung getragen, daß die Objekte des Grundbuchs Bezeichner von Objekten sind.

5.3.2 Relationen

Relationen werden danach eingeteilt, ob sie das formelle Recht abbilden oder das materielle. Die Modellierung beider Arten erfolgt als situationsabhängige Prädikate, als Fluents. Zur Erweiterung der Beschreibung des formellen Rechts im Rahmen des Grundbuchverfahrens wird auf Abschnitt 5.3.4 verwiesen. Neben Relationen, die sich situationsabhängig verändern, kann es auch unveränderliche Relationen geben.

- Relationen, die das formelle Recht betreffen:

  Dokument $\subseteq$ *Grundstück_ID* x *Person_ID* x *Person_ID* x *Situations*.
  Dokument(g,p1,p2,r,s) bezeichnet ein Dokument, das das Grundstück g betrifft und in der Situation s zwischen den Personen p1 und p2 über das Recht r abgeschlossen wurde.
  Ein Dokument dient als Beweis für ein bestimmtes Recht.

- Relationen, die das materielle Recht betreffen:

  Eingetragenes_Recht $\subseteq$ *Grundstück_ID* x *Person_ID* x *Rechte* x *Situations*
  Eingetragenes_Recht(g,p,r,s) bezeichnet einen Grundbucheintrag über ein Recht r zugunsten einer Person p, die das Grundstück g betrifft.
  Wert_Recht $\subseteq$ *Grundstück_ID* x *Person_ID* x *Rechte* x *Reals* x *Situations*,
  Wert_Recht(g,p,r,w,s) bezeichnet den Wert des eingetragenen Rechtes *Eingetragenes_Recht(g,p,r,s)*. Einen Wert hat zum Beispiel eine Hypothek.
  Grundstück_Flurstück(g,f,s) $\subseteq$ *Grundstück_ID* x *Flurstück_ID* x *Situations*
  Hiermit wird bezeichnet, daß das Flurstück f Bestandteil des Grundstücks g ist.

- situationsunabhängige Relationen:

Rechte unterschieden sich danach, ob es Vollrechte oder Teilrechte (siehe Abschnitt 2.7.1.1) sind und ob das Teilrecht ein Nutzungsrecht ist oder ein Recht, das den Verkaufswert des Grundstücks beschränkt. In diesem Modell ist ein Vollrecht ein *Eigentumsrecht*, und Teilrechte sind die *Hypothek* und der *Nießbrauch* als Vertreter der zwei Arten der Teilrechte. Dies drücken die folgenden situationsunabhängigen Prädikate aus:

Vollrecht(x)$\Leftrightarrow$*x=Eigentum*
Teilrecht_Wert(x)$\Leftrightarrow$*x=Hypothek*
Teilrecht_Nutzung(x)$\Leftrightarrow$*x=Nießbrauch*

5.3.3 Die Beschreibung der elementaren Operationen

Elementare Operationen werden entsprechend dem im Kapitel 4 beschriebenen Ansatz als Transaktionen beschrieben. Der letzte Parameter bezeichnet die Transaktionszeit, also die Zeit, zu der die Transaktion abläuft.

Es sollen folgende Operationen möglich sein:

- externe Transaktionen:

  register_Dokument: Grundstück_ID x *Person_ID* x *Person_ID* x *Rechte* x *Integers* →*Transactions*,
  Dokumente, die als Beweis für ein Recht gelten, sollen registriert werden können.
  register_Dokument(g,p1,p2,r,t) bezeichnet den Vorgang, daß ein Dokument über den Transfer eines Rechts r an einem Grundstück g von Person p1 auf Person p2 registriert wird.

- interne Transaktionen:

  eintragen_Vollrecht: Grundstück_ID x *Personen_ID* x *Rechte* x *Integers* →*Transactions*
  eintragen_Vollrecht(g,p,r,t) bezeichnet die Eintragung eines Rechts r an einem Grundstück g zu Gunsten einer Person p. Bei diesem Recht handelt es sich um ein Vollrecht, also um das Eigentumsrecht.
  löschen_Vollrecht: Grundstück_ID x *Personen_ID* x *Rechte* x *Integers* → *Transactions*
  löschen_Vollrecht(g,p,r,t) bezeichnet die Löschung eines Rechts r, das zu Gunsten einer Person p an einem Grundstück g eingetragen war. Bei diesem Recht handelt es sich um ein Vollrecht, also um das Eigentumsrecht.
  eintragen_Teilrecht_Nutzung: Grundstück_ID x *Person_ID* x *Rechte* x *Integers* → *Transactions*
  eintragen_Teilrecht_Nutzung(g,p,r,t) bezeichnet die Eintragung eines Nutzungsrechts r an einem Grundstück g zu Gunsten von Person p an einem Grundstück g. Dies ist in diesem Modell ein Nießbrauch.
  löschen_Teilrecht_Nutzung: Grundstück_ID x *Person_ID* x *Rechte* x *Integers* → *Transactions*
  löschen_Teilrecht_Nutzung(g,p,r,t) bezeichnet die Löschung eines Nutzungsrechts r, das zu Gunsten von Person p an einem Grundstück g eingetragen war. Dies ist in diesem Modell ein Nießbrauch.
  eintragen_Teilrecht_Wert: Grundstück_ID x *Person_ID* x *Rechte* x *Reals* x *Integers* → *Transactions*
  eintragen_Teilrecht_Wert(g,p,r,w,t) bezeichnet die Eintragung eines Rechts r mit dem Wert w zu Gunsten von Person p an einem Grundstück g. In diesem Modell handelt es sich dabei um eine Hypothek.
  löschen_ Teilrecht_Wert: Grundstück_ID x *Person_ID* x *Rechte* x *Reals* x *Integers* → *Transactions*
  löschen_Teilrecht_Wert(g,p,r,w,t) bezeichnet die Löschung eines Rechts r mit dem Wert w, das zu Gunsten von Person p an einem Grundstück g eingetragen war. In diesem Modell handelt es sich dabei um eine Hypothek.
  eintragen_vereinigen_Grundstück: Grundstück_ID x *Grundstück_ID* x *Integers* →*Transactions*,
  eintragen_vereinigen_Grundstück(g1,g2,t) bezeichnet den Vorgang der Eintragung der Vereinigung zweier Grundstücke g1 und g2. Das Grundstück g2 hört dabei auf zu bestehen.
  eintragen_teilen_Grundstück: Grundstück_ID x *Grundstück_ID* x *Flurstück_ID* x *Integers* →*Transactions*,

eintragen_teilen_Grundstück(g1,g2,f,t) bedeutet die Eintragung der Teilung des Grundstücks g1. f bezeichnet das Flurstück, das aus dem Bestand von g1 herausgelöst und danach als g2 geführt wird.

5.3.3.1 Vorbedingungen für Transaktionen: Transaction Precondition Axiome

- externe Transaktionen:

  Poss(register_Dokument(g,p1,p2,r,t),s) ⇔ ¬*Dokument(g,p1,p2,r,s)*
  Ein Dokument kann registriert werden, wenn es noch nicht registriert ist.

- interne Transaktionen:

  Poss(eintragen_Vollrecht(g,p,r,t),s) ⇔ *Vollrecht(r)*∧ ¬*Eingetragenes_Recht(g,p,r,s)*
  Ein Vollrecht ist ein Eigentumsrecht und kann eingetragen werden, wenn es noch nicht eingetragen ist.
  Poss(löschen_Vollrecht(g,p,r,t),s) ⇔ *Vollrecht(r)*∧ *Eingetragenes_Recht(g,p,r,s)*
  Ein Vollrecht kann gelöscht werden, wenn es im Grundbuch eingetragen ist.
  Poss(eintragen_Teilrecht_Nutzung(g,p,r,t),s) ⇔ *Teilrecht_Nutzung(r)*∧ ¬*Eingetragenes_Recht(g,p,r,s)*
  Ein Nutzungsrecht kann zu Gunsten einer Person eingetragen werden, wenn es noch nicht eingetragen ist.
  Poss(löschen_Teilrecht_Nutzung(g,p,r,t),s) ⇔ *Eingetragenes_Recht(g,p,r,s)*∧ *Teilrecht_Nutzung(r)*
  Ein Nutzungsrecht kann gelöscht werden, wenn es im Grundbuch eingetragen ist.
  Poss(löschen_Teilrecht_Wert(g,p,r,w,t),s) ⇔ *Eingetragenes_Recht(g,p,r,s)*∧ *Teilrecht_Wert(r)*∧*Wert_Recht(g,p,r,w,t)*
  Ein Teilrecht, das einen Wert hat, kann gelöscht werden, wenn es zusammen mit diesem Wert im Grundbuch eingetragen ist.
  Poss(eintragen_Teilrecht_Wert(g,p,r,w,t),s) ⇔ *Teilrecht_Wert(r)*∧*Vollrecht(r')*∧*w>0*∧ ¬*Eingetragenes_Recht(g,p,r,s)*
  Ein Teilrecht kann zu Gunsten einer Person im Grundbuch eingetragen werden, wenn es noch nicht eingetragen ist und der Wert des Rechts größer als 0 ist.
  Poss(eintragen_vereinigen_Grundstück(g1,g2,t),s) ⇔ ∃*p*∃*r* *Eingetragenes_Recht(g1,p,r,s)*∧ *Eingetragenes_Recht(g2,p,r,s)*∧ *Vollrecht(r)*
  Die Eintragung der Vereinigung zweier Grundstücke kann geschehen, wenn sie denselben Eigentümer haben, es sich also um eine Vereinigung im eigenen Besitz handelt.
  Poss(eintragen_teilen_Grundstück(g1,g2,f,t),s) ⇔ ∃*p*∃*r* *eingetragenes_Recht(g1,p,r,s)*∧*Vollrecht(r)*∧ *Grundstück_Flurstücke(g1,fl,s)*∧∃*f2 Grundstück_Flurstücke(g1,f2,s)*∧*f*≠*f2*
  Die Teilung im eigenen Besitz kann vorgenommen werden, wenn das abzuteilende Flurstück Teil des zu teilenden Grundstücks ist und sich nach der Teilung noch Flurstücke im Bestand dieses Grundstücks befinden.

5.3.3.2 Die Wirkungen von Transaktionen: Successor State Axiome

- Relationen, die das formelle Recht betreffen:

Poss(a,s) ⇒ (Dokument(g,p1,p2,r,do(a,s)) ⇔
a=register_Dokument(g,p1,p2,r,t)∨ Dokument(g,p1,p2,r,s))
Ein Dokument ist im Grundbuch als Beweis verfügbar, wenn es entweder gerade registriert wird oder bereits im Grundbuch registriert ist.

- Relationen, die das materielle Recht betreffen:

Poss(a,s) ⇒ (Eingetragenes_Recht(g,p,r,do(a,s)) ⇔
(a=eintragen_Vollrecht(g,p,r,t)∨ a=eintragen_Teilrecht_Wert(g,p,r,w,t)
∨a=eintragen_Teilrecht_Nutzung(g,p,r,t))∨ Eingetragenes_Recht(g,p,r,s)∧
¬a=löschen_Vollrecht(g,p,r,t)∧ ¬a=löschen_Teilrecht_Wert(g,p,r,w',t)∧
¬a=löschen_Teilrecht_Nutzung(g,p,r,t))
Zu Gunsten einer Person besteht ein Recht an einem Grundstück, wenn es gerade eingetragen wurde oder bereits eingetragen war und nicht gerade gelöscht wurde.
Poss(a,s) ⇒ (Wert_Recht(g,p,r,w,do(a,s)) ⇔
a=eintragen_Teilrecht_Wert(g,p,r,w,t)∨
Wert_Recht(g,p,r,w,s)∧ ¬a=löschen_Teilrecht_Wert(g,p,r,w,t))
Für ein Teilrecht ist ein Wert für dieses Recht eingetragen, wenn es entweder gerade eingetragen wurde oder bereits eingetragen war und nicht gelöscht wurde.
Poss(a,s) ⇒ (Grundstück_Flurstücke(g,f,do(a,s)) ⇔
a=eintragen_vereinigen_Grundstück(g,g',p',t) ∧ Grundstück_Flurstücke(g',f,s)
∨ Grundstück_Flurstücke(g,f,s) ∧¬a=eintragen_teilen_Grundstück(g,g'',p'',f,t)
∧¬a=eintragen_vereinigen_Grundstück(g'',g,p''',t))
Ein Flurstück ist Bestandteil des Grundstücks g, wenn es g durch Eintragung einer Vereinigung gerade zugeordnet wurde oder bereits als Bestandteil dieses Grundstücks eingetragen war und nicht von ihm abgeteilt wurde und das Grundstück nicht nach Eintragung einer Vereinigung mit einem anderen Grundstück aufgehört hat zu bestehen.

5.3.3.3 Die initiale Datenbank

Die initiale Datenbank enthält die situationsunabhängigen Relationen und die Definition von *time* für alle Transaktionen.

Vollrecht(x)⇔x=Eigentum
Teilrecht_Wert(x)⇔x=Hypothek
Teilrecht_Nutzung(x)⇔x=Nießbrauch
time(register_Dokument(g,p1,ps,r,t))=t
time(eintragen_Vollrecht(g,p,r,t))=t
time(löschen_Vollrecht(g,p,r,t))=t
time(eintragen_Teilrecht_Wert(g,p,r,w,t))=t
time(löschen_Teilrecht_Wert(g,p,r,w,t))=t
time(eintragen_Teilrecht_Nutzung(g,p,r,t))=t
time(löschen_Teilrecht_Nutzung(g,p,r,t))=t
time(eintragen_vereinigen_Grundstück(g1,g2,t))=t
time(eintragen_teilen_Grundstück(g1,g2,fl,t))=t

5.3.4 Die Beschreibung der komplexen Abläufe

Neben der Beschreibung der elementaren Operationen als Transaktionen ist es nun notwendig, das Modell zu erweitern, damit komplexe Abläufe modelliert werden können.

Grundbuchverfahren müssen als komplexe Transaktionen beschrieben werden. Was sind nun aber für Erweiterungen des Modells notwendig? Entsprechend dem konzeptionellen Modell steht am Anfang eines Grundbuchverfahrens der Antrag. Es müssen folglich Anträge im System integriert werden. Zur Beschreibung des Verfahrens ist der Prozeßgedanke aus 4.6.1 hilfreich. Ein Eintragungsverfahren ist ein Prozeß und besteht aus einer Anfangs- und einer Endtransaktion. Transaktionen auf den Grundbuchdaten dürfen nur innerhalb eines solchen Prozesses stattfinden. Um Inkonsistenzen zu vermeiden, kann nur ein Grundbuchverfahren zu einer Zeit an einem Grundstück ablaufen. Zur Beschreibung des Grundbuchverfahrens müssen also folgende Aufgaben gelöst werden:

1. Die Integration von Anträgen,
2. die Formalisierung des Eintragungsprozeßes, sowie
3. die Beschreibung der einzelnen Grundbuchverfahren.

5.3.4.1 Anträge

Anträge werden als Fluents beschrieben, also als Relationen, die sich situationsabhängig verändert. Anträge, die im System gespeichert sind, bezeichnen die derzeit im Grundbuchamt zu bearbeitenden Anträge auf Eintragung von Rechten im Grundbuch. Aus diesen Anträgen muß ersichtlich sein, was das Ziel des Verfahrens ist, zum Beispiel die Eintragung eines Eigentumstransfers oder einer Grundstücksteilung. Notwendig ist die Angabe des oder der Grundstücke, die betroffen sind, sowie der beteiligten Personen. Bei Grundstücksteilungen ist die Angabe des zu teilenden Grundstücksteils notwendig durch Angabe der Flurstücksnummer.
Die Sprache des Situationskalküls wird erweitert um eine neue Sorte: *Anträge*. Die intendierte Bedeutung ist die Angabe der Art des gestellten Antrags. Diese Art ist abhängig vom Ziel des angestrebten Grundbuchverfahrens, zum Beispiel kann das Ziel eines Grundbuchverfahrens die Eintragung eines Eigentumstransfers sein. Im einzelnen existieren also folgende Objekte der Sorte *Anträge*:

Eigentumstr_e, Hypothek_l, Hypothek_e, Nießbrauch_e, Teilung_e,
Vereinigung_e.

Es sind entsprechend dieser Reihenfolge folgende Anträge möglich:

- Antrag auf Eigentumstransfer,
- Antrag auf Löschung einer Hypothek,
- Antrag auf Eintragung einer Hypothek,
- Antrag auf Eintragung eines Nießbrauchs,
- Antrag auf die Eintragung der Teilung eines Grundstücks,
- Antrag auf die Eintragung der Vereinigung zweier Grundstücke.

Die Menge der Fluents wird erweitert durch:

Antrag $\subseteq$ *Anträge* x *Grundstück_ID* x *Grundstück_ID* x *Flurstück_ID* x *Person_ID* x *Person_ID* x *Reals* x *Situations*

Antrag(a,g1,g2,fl,p1,p2,w,s) ist ein Antrag der Art a, der die Grundstücke g1 und g2 und das Flurstück fl betrifft. Der Antrag betrifft die Personen p1 und p2, die an dem beantragten Grundbuchverfahren beteiligt sind. Der Wert des einzutragenden Rechts ist w.

5.3.4.2 Der Eintragungsprozeß

Den Eintragungsprozeß rahmen eine Anfangs- und eine Endtransaktion ein. Ein Fluent zeigt an, daß an einem Grundstück bereits ein Grundbuchverfahren läuft.
Ein Eintragungsprozeß soll möglich sein, wenn bezüglich des Grundstücks, das er betrifft, noch kein anderer Eintragungsprozeß läuft.

Die Beschreibung des Eintragungsprozesses erfolgt wie in Beispiel 4.1.
Das Grundbuchverfahren wird eingeleitet von den Transaktionen *start_Grundbuchverfahren* und beendet durch *end_Grundbuchverfahren*. Das Fluent *Laufend_Grundbuchverfahren* zeigt an, daß an einem Grundstück bereits ein Grundbuchverfahren läuft.

- Relation:

  Laufend_Grundbuchverfahren ⊆ *Grundstück_ID* x *Situations*
  Laufend_Grundbuchverfahren(g,s) sagt aus, das in der Situation s ein Eintragsverfahren läuft, das das Grundstück g betrifft.

- Transaktionen:

  start_Grundbuchverfahren: Grundstück_ID x *Integers* → *Transactions*
  start_Grundbuchverfahren(g,t) bezeichnet den Anfang eines Eintragungsverfahrens, der das Grundstück g betrifft und zur Zeit t stattfindet.
  end_Grundbuchverfahren: Grundstück_ID x *Integers* → *Transactions*
  end_Grundbuchverfahren(g,t) bezeichnet das Ende eines Eintragungsverfahrens, der das Grundstück g betrifft und zur Zeit t stattfindet.

- Vorbedingungen für Transaktionen des Eintragungsprozesses: Transaction Precondition Axiome:

  Poss(start_Grundbuchverfahren(g,t),s) ⇔ ¬*Laufend_Grundbuchverfahren(g,s)*
  Ein Grundbuchvefahren an einem Grundstück soll möglich sein, wenn noch kein anderes läuft.
  Poss(end_Grundbuchverfahren(g,t),s) ⇔ *Laufend_Grundbuchverfahren(g,s)*
  Nur ein laufendes Grundbuchverfahren kann beendet werden.

- Das Successor State Axiom für das Fluent *Laufend_Grundbuchverfahren(g,s)*

  Poss(a,s) ⇒ *(Laufend_Grundbuchverfahren(g,do(a,s))* ⇔
  a=start_Grundbuchverfahren(g,t)∨*Laufend_Grundbuchverfahren(g,s)*∧
  ¬*a=end_Grundbuchverfahren(g,t))*
  Ein Grundbuchverfahren läuft an einem Grundstück, wenn es gerade begonnen wurde oder bereits läuft und nicht beendet wird.

5.3.4.3 Die Grundbuchverfahren

Entsprechend dem konzeptionellen Modell können nun die Abläufe in der Sprache der komplexen Transaktionen formuliert werden.
Grundbuchverfahren stellen die Verbindung zwischen materiellem Recht und seiner Abbildung im Grundbuch her. Sie setzen die Vorschriften des formellen Rechts um.
Zur formalen Beschreibung wird die Funktion *now* eingeführt. Sie soll den aktuellen Zeitpunkt liefern. Die Spezifikation dieser Funktion wird offen gelassen, ist es doch ein Problem der Implementation, wie der aktuelle Zeitpunkt bestimmt wird. Es wird hierbei verwiesen auf Kapitel 7.
In der Spezifikation der Grundbuchverfahren kommen externe Transaktionen nicht explizit vor. Nur die von ihnen beeinflußten Relationen - das sind genau die, die das formelle Recht abbilden - werden genutzt und bilden den Beweis für materielle Rechtsänderungen. Wie und wann diese Beweise in das Grundbuch eingebracht werden, kann nicht Bestandteil der Spezifikation sein, da es sich um Handlungen von Personen in der materiellen Welt handelt.

- Verfahren zur Eintragung eines Eigentumstransfers:

  proc Transfer_Eigentum (g1,p1,p2)
  ?(∃x,y,z Antrag(transfer_e,g1,x,y,p1,p2,z));
  start_Grundbuchverfahren(g1,now);
  ?(Eingetragenes_Recht(p1,g1,Eigentum));
  ?(Dokument(g1,p1,p2,Eigentum));

πr (eintragen_Vollrecht(g1,p2,r);löschen_Vollrecht(g1,p1,r));
end_Grundbuchverfahren(g1,now)

endProc

Bei Vorliegen eines Antrags auf Eigentumstransfer wird das Grundbuchverfahren eröffnet. Es wird geprüft, ob der Verkäufer Eigentümer des Grundstücks ist. Dann wird der Eigentumstransfer vorgenommen. Dazu erfolgt die Eintragung des Eigentumsrechts des neuen Eigentümers und die Löschung des Eigentumsrechts des alten Eigentümers.

- Verfahren zur Eintragung einer Hypothek:

proc eintragen_Hypothek (g,p,w)
?(∃x,y,z Antrag(Hypothek_e,g,x,y,p,z,w));
start_Grundbuchverfahren(g,now);
?(∃p' Dokument(g,p',p,Hypothek)∧Eingetragenes_Recht(g,p',Eigentum));
eintragen_Teilrecht_Wert(g,p,Hypothek,w,now);
end_Grundbuchverfahren(g,now)

endProc

Liegt ein Antrag auf Eintragen einer Hypothek vor, dann wird das Grundbuchverfahren eingeleitet. Es wird geprüft, ob ein Dokument vorliegt, das das Zustandekommen dieser Hypothek beweist. Dieses Dokument muß mit dem Eigentümer des Grundstücks abgeschlossen sein. Ist dies der Fall, wird die Hypothek eingetragen und das Grundbuchverfahren beendet.

- Verfahren zum Löschen einer Hypothek:

proc löschen_Hypothek (g,p,w)
?(∃x,y,z Antrag(Hypothek_l,g,x,y,p,z,w));
start_Grundbuchverfahren(g,now);
?(∃p' Dokument(g,p',p,Hypothek)∧Eingetragenes_Recht(g,p',Hypothek)∧
Wert_Recht(g,p',r,w)∧Eingetragenes_Recht(g,p,Eigentum));
löschen_Teilrecht_Wert(g,p,w,now);
end_Grundbuchverfahren(g,now);

endProc;

Bei Vorliegen eines Antrags auf Löschung einer Hypothek wird das Grundbuchverfahren eröffnet. Es wird geprüft, ob ein Dokument die Einigung zwischen dem Gläubiger der Hypothek und dem Eigentümer des Grundstücks beweist. Ist dies der Fall, wird die Hypothek gelöscht und das Grundbuchverfahren beendet.

- Verfahren zur Eintragung eines Nießbrauches:

proc eintragen_Nießbrauch (g,p)
?(∃x,y,z,z' Antrag(Nießbrauch_e,g,x,y,p,z,z'));
start_Grundbuchverfahren(g,now);
?(∃p' Dokument(g,p',p,Nießbrauch)∧Eingetragenes_Recht(g,p',Eigentum));
eintragen_Teilrecht_Nutzung(g,p,Nießbrauch,now);
end_Grundbuchverfahren(g,now)

endProc

Liegt ein Antrag auf Eintragen eines Nießbrauches vor, dann wird das Grundbuchverfahren eingeleitet. Es wird geprüft, ob ein Dokument vorliegt, das das Zustandekommen dieses Nießbrauches beweist. Dieses Dokument muß mit dem Eigentümer des Grundstücks abgeschlossen sein. Ist dies der Fall, wird der Nießbrauch eingetragen und das Grundbuchverfahren beendet.

- Verfahren zur Eintragung einer Grundstücksvereinigung:

proc vereinigen_Grundstück (g1,g2,p)
?(∃x,y,z Antrag(Vereinigung_e,g1,g2,x,p,y,z));
start_Grundbuchverfahren(g1,now);
start_Grundbuchverfahren(g2,now);
?(Eingetragenes_Recht (g1,p,Eigentum)∧
Eingetragenes_Recht (g2,p,Eigentum));
πg3 eintragen_vereinigen_Grundstück(g1,g2,g3,now);
start_Grundbuchverfahren(g3, now);
while (∃r ∃p' eingetragenes_Recht(g2,p',r))
(πp'πr (?Eingetragenes_Recht(g2,p',r)∧Vollrecht(r));
löschen_Vollrecht(g',p',r,now)) |
(πp'πr (?Eingetragenes_Recht(g2,p',r)∧Teilrecht_Nutzung(r));
löschen_Teilrecht_Nutzung(g2,p',r,now);
eintragen_Teilrecht_Nutzung(g1,p',r,now)) |
(πp'πrπw (?Eingetragenes_Recht(g2,p',r)∧Teilrecht_Wert(r)∧
Wert_Recht(g2,p',r,w));
löschen_Teilrecht_Wert(g2,p',r,w,now);
eintragen_Teilrecht_Wert(g1,p',r,w,now))
endWhile;
end_Grundbuchverfahren(g1,now);
end_Grundbuchverfahren(g2,now)
endProc

Wenn ein Antrag auf Grundstücksvereinigung vorliegt, wird das Grundbuchverfahren eröffnet. Wenn die antragstellende Person Eigentümer beider Grundstücke ist, dann wird die Vereinigung vorgenommen. Bei der Vereinigung werden alle Rechte von dem Nebengrundstück auf das Hauptgrundstück übertragen.

- Verfahren zur Eintragung einer Grundstücksteilung:

proc teilen_Grundstück (g1,g2,fl,p)
?(∃x,y,z Antrag(Teilung_e,g1,x,fl,p,y,z));
start_Grundbuchverfahren(g1,now);
?(Grundstück_Flurstücke(g1,fl));
?(Eingetragenes_Recht (g1,p,Eigentum));
eintragen_teilen_Grundstück(g1,g2,fl,now);
start_Grundbuchverfahren(g2,now);
eintragen_Vollrecht(g2,p,Eigentum,now);
end_Grundbuchverfahren(g2,now);
end_Grundbuchverfahren(g1,now);
endProc

Nach Vorliegen eines Antrags auf Grundstücksteilung wird das Grundbuchverfahren eröffnet. Es wird geprüft, ob das abzuteilende Flurstück auch Teil des Ausgangsgrundstücks ist und ob die antragstellende Person auch Eigentümer dieses Grundstücks ist. Dann wird die Grundstücksteilung vorgenommen. Zu Gunsten der antragstellenden Person wird das Eigentumsrecht am neu gebildeten Grundstück eingetragen.

5.4 Das Modell der Beziehungen zwischen materiellem Recht und Grundbuch

In welcher Beziehung stehen nun das Modell des materiellen Rechts und das Modell des Grundbuchs? Was muß Inhalt des Grundbuchs sein, wenn in der materiellen Welt bestimmte Veränderungen stattfinden? Das Legalitätsprinzip besagt, daß Grundbuch und materielles Recht in Übereinstimmung stehen müssen. Ist dies nicht der Fall, so ist das Grundbuch unrichtig. Was bedeutet es bezogen auf das Modell, daß materielles Recht und Grundbuch in Übereinstimmung stehen? Wenn ein materielles Recht an einem Grundstück besteht, dann muß auch eine Eintragung über dieses Recht im Grundbuch vorhanden sein. Bezüglich der Formalisierung im Situationskalkül bedeutet das, daß Fluents, die materielles Recht abbilden im Modell des Grundbuches, gleichzeitig mit Fluents aus dem Modell des materiellen Rechts gelten. Zum Beispiel gilt das Fluent *Eingetragenes_Recht(G1,P1,Eigentum,s)* des Grundbuchs, genau dann, wenn das Fluent *Eigentümer(G1,P1,s)* gilt in einer Situation s. Diese Idee wird Grundlage der Modellierung von Inkonsistenzmarkierungen sein. Inkonsistenzmarkierungen sind Fluents, die genau dann gelten, wenn Fluents des Grundbuchs die materielle Rechtslage unrichtig abbilden. Das heißt, wenn Fluents des materiellen Rechts und des Grundbuchs nicht gleichzeitig gelten, obwohl dies der Fall sein müßte. Inkonsistenzmarkierungen werden eingetragen, wenn die Bedingungen für die Richtigkeit des Grundbuchs nicht eingehalten werden. Diese Bedingungen werden Konsistenzbedingungen genannt und bilden die Grundlage für die Markierung der Inkonsistenzen. Die Konsistenzbedingungen werden im folgenden beschrieben.

5.4.1 Konsistenzbedingungen

Konsistenzbedingungen zeigen an, ob sich Grundbuch und materielle Rechtsverhältnisse in Übereinstimmung befinden. Sie beziehen sich einerseits auf die Rechte an Grundstücken und andererseits auf die räumlichen Verhältnisse. Die Rechtsverhältnisse, die in diesem Modell vorkommen (Eigentumsrecht, Hypothek, Nießbrauch), müssen mit den entsprechenden Instanzen des Fluents *Eingetragenes_Recht* übereinstimmen. Die Übereinstimmung der räumlichen Verhältnisse ist gegeben durch die Zuordnung der Flurstücke zu den Grundstücken. Entsprechend müssen die Fluents *Grundstück_Flurstücke* und *Grundstück_Flurstücke_m* übereinstimmen. Formal ausgedrückt bedeutet das:

- Übereinstimmung der Eigentumsrechte:
  $Eigentümer(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Eigentum,s)$
- Übereinstimmung der hypothekarischen Belastungen:
  $Gläubiger\_Hypothek(g,p,w,s) \Leftrightarrow$
  $Eingetragenes\_Recht(g,p,Hypothek,s) \wedge Wert\_Recht(g,p,Hypothek,w,s)$
- Übereinstimmung der Nießbrauchrechte:
  $Nießbraucher(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Nießbrauch,s)$
- Übereinstimmung der räumlichen Gegebenheiten:
  $Grundstück\_Flurstücke\_m(g,fl,s) \Leftrightarrow Grundstück\_Flurstücke(g,fl,s)$

5.4.2 Inkonsistenzmarkierungen

Aus diesen Konsistenzbedingungen können nun Fluents abgeleitet werden, die die Inkonsistenzen markieren. Inkonsistent sind Eintragungen dann, wenn die Konsistenzbedingungen nicht eingehalten werden. Diese Fluents werden wie folgt definiert:

- inkonsistenter Eintrag über ein Eigentumsrecht:
  $Inkons\_Eigentümer(g,p,s) \Leftrightarrow$
  $\neg(Eigentümer(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Eigentum,s))$

Inkons_Eigentümer ist ein definiertes Fluent oder aus der Datenbanksicht betrachtet ein View (Reiter 1995; Arenas und Bertossi 1998). In (Reiter 1995) ist beschrieben, wie eine solche Definition in eine Updatespezifikation umgewandelt wird. Für den Zweck des formalen Modells ist die Tatsache ausreichend, daß diese Definitionen in eine zur Datenbankspezifikation passende Form übersetzt werden kann. Die weiteren Inkonsistenzmarkierungen werden wie folgt definiert:

- inkonsistenter Eintrag über eine Hypothek:

  $$Inkons\_Hypothek(g,p,w,s) \Leftrightarrow \neg(Gläubiger\_Hypothek(g,p,w,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Hypothek,s) \wedge Wert\_Recht(g,p,Hypothek,w,s))$$

- inkonsistenter Eintrag über einen Nießbrauch:

  $$Inkons\_Nießbrauch(g,p,s) \Leftrightarrow \neg(Nießbraucher(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Nießbrauch,s))$$

- inkonsistente räumliche Gegebenheiten:

  $$Inkons\_Grundstück\_Flurstücke(g,fl,s) \Leftrightarrow \neg(Grundstück\_Flurstücke\_m(g,fl,s) \Leftrightarrow Grundstück\_Flurstücke(g,fl,s))$$

5.4.3 Aussagen über das Verhältnis von materieller Rechtslage und Grundbuch

Ziel der Modellierung der Beziehungen zwischen dem Modell der materiellen Rechtslage und dem Modell des Grundbuchs war es, Aussagen über das Verhältnis beider, insbesondere über den Grad der Übereinstimmung zu treffen. Die vollständige Übereinstimmung ist der ideale Fall, der durch die Modellierung des Grundbuchs allein nicht erreicht werden kann. Die Gründe hierfür wurden im Zusammenhang mit der Grundbuchunrichtigkeit (siehe Abschnitt 2.8) diskutiert. Eine Abbildung des Grundbuchs selbst läßt maximal Aussagen über das Verhältnis von Grundbuch und seiner Repräsentation zu, nicht aber über Inkonsistenzen, die aus der Nichtübereinstimmung von Grundbuch und materieller Welt erwachsen. Durch die Modellierung des Grundbuchs und der materiellen Rechtslage können nun Aussagen darüber erfolgen, wie sich die beiden Modelle zueinander verhalten. Damit ist das Problem nicht im allgemeinen Fall gelöst, beschränken sich doch die Aussagen auf das Modell der materiellen Welt und beziehen sich nicht auf die Welt selbst. Trotzdem können verschiedene Probleme jetzt besser betrachtet werden. Das sind Probleme, die aus der Unvollständigkeit der Repräsentation der materiellen Rechtsverhältnisse durch das Grundbuch erwachsen, wie sie in Abschnitt 3.5.3 beschrieben wurden. An dieser Stellen sollen nun einige Beispiele angegeben werden, die aufzeigen, welche Fragen sich in diesem Ansatz nun diskutieren lassen. Zu einer ausführlichen Diskussion wird auf das Kapitel 7 verwiesen.

- Verwechslungen bezüglich des Rechtsgegenstands:

Zwei Personen einigen sich über den Verkauf eines Teils eines Grundstücks. Beide stimmen bei dem Verkaufsgegenstand, also dem Flurstück, überein. Es passiert aber, daß die Nummer des Flurstücks bei der Eintragung im Grundbuch verwechselt wird. Bei der Eintragung wird der Fehler nicht bemerkt, da sich das irrtümlich bezeichnete Flurstück auch im Eigentum des Verkäufers befindet. Die Eintragung wird also vorgenommen. Das Grundbuch wird unrichtig, da die materiellrechtlichen Voraussetzungen nicht vorlagen, denn die bezogen sich auf ein anderes Flurstück.

- Rechtsveränderungen außerhalb des Grundbuchs:

Rechtsveränderungen geschehen außerhalb des Grundbuchs zum Beispiel durch den Tod einer Person. Eine Person stirbt, zu deren Gunsten ein Nießbrauch im Grundbuch eingetragen ist. Ein Nießbrauch ist auf die Lebenszeit der Person beschränkt, deshalb wird das Grundbuch mit diesem Ereignis unrichtig. Dies liegt daran, daß zwar die Person im Modell der materiellen Rechtslage nicht mehr existiert, aber die Eintragung im Grundbuch über diesen Nießbrauch erhalten bleibt. Diese Unrichtigkeit ist nun aber über die Inkonsistenzmarkierung für den Nießbrauch markiert.

- Nichteinhaltung der Verfahrensregeln:

Eine Person kauft von einer anderen Person ein Grundstück. Sie versäumt es, ihr Recht im Grundbuch eintragen zu lassen. Daraufhin wird das Grundbuch unrichtig, da der Eigentumstransfer materiellrechtlich zustande gekommen ist, dies aber nicht aus dem Grundbuch hervorgeht. Da die Eintragung im Grundbuch aber zu Gunsten eines Dritten als richtig gilt, besteht die Gefahr des gutgläubigen Erwerbs. Der ursprüngliche Eigentümer könnte das Grundstück noch einmal verkaufen.

5.4.4 Die Korrektur des unrichtigen Grundbuchs

Es wurden die Grundlagen geschaffen, Inkonsistenzen im Grundbuch zu markieren. Ein wichtiger Schritt fehlt noch: die Beschreibung der Transaktionen, die das Grundbuch korrigieren. Das Grundbuchamt ist verpflichtet, die Richtigkeit des Grundbuchs zu erhalten und somit die Korrektur des Grundbuchs zu veranlassen, wenn eine solche Unrichtigkeit bekannt wird. Bekannt gemacht werden kann dies durch eine Beschwerde. Für das formale Modell können also korrigierende Maßnahmen für den Fall beschrieben werden, daß eine Beschwerde gegen die Richtigkeit des Grundbuchs vorliegt und Inkonsistenzen markiert wurden. Die verbleibenden Aufgaben sind nun, die Verarbeitung von Beschwerden und die Abläufe zur Korrektur des Grundbuchs zu beschreiben. Dieses Modell beschränkt sich auf die Beschreibung der Korrektur inkonsistenter Rechtsverhältnisse.

5.4.4.1 Beschwerden

Zur formalen Beschreibung von Beschwerden wird das Fluent *Beschwerde* eingeführt, sowie weiterhin eine Sorte *Beschwerdeart*. Es gibt zwei Objekte der Sorte *Beschwerdeart*: *eintragen* und *löschen*. Damit wird der Gedanke umgesetzt, daß Beschwerden gegen eine überflüssige und somit zu löschende Eintragung oder gegen eine fehlende Eintragung geführt werden können. Beschwerden können *registriert (also eingetragen)* und nach ihrer Bearbeitung *gelöscht* werden.

- Fluents

  Beschwerde $\subseteq$ *Grundstück_ID* x *Person_ID* x *Rechte* x *Reals* x *Beschwerdeart* x *Situations*

  Beschwerde(g,p,r,w,art,s) bezeichnet die Tatsache, daß in der Situation s eine Beschwerde der Sorte art registriert ist, die das Recht r am Grundstück g mit dem Wert w betrifft.

- Transaktionen

  register_Beschwerde: Grundstück_ID x *Person_ID* x *Rechte* x *Reals* x *Beschwerdeart* x *Integers* → *Transactions*

  register_Beschwerde(g,p,r,w,art,t) bezeichnet den Vorgang, daß zum Zeitpunkt t eine Beschwerde beim Grundbuchamt registriert wird.

  löschen_Beschwerde: Grundstück_ID x *Person_ID* x *Rechte* x *Reals* x *Beschwerdeart* x *Integers* → *Transactions*

  löschen_Beschwerde(g,p,r,w,art,t) bezeichnet den Vorgang, daß zum Zeitpunkt t eine Beschwerde gelöscht wird.

- Vorbedingungen für die Transaktionen des Beschwerdeverfahrens

  $Poss(register\_Beschwerde(g,p,r,w,art,t)) \Leftrightarrow \neg laufend\_Grundbuchverfahren(g,s)$

  Eine Beschwerde gegen die Richtigkeit des Grundbuchs soll möglich sein, wenn sie kein laufendes Grundbuchverfahren betrifft.

  $Poss(loeschen\_Beschwerde(g,p,r,w,art,t)) \Leftrightarrow Beschwerde(g,p,r,w,art,s)$

  Eine Beschwerde kann nur gelöscht werden, wenn sie auch vorhanden ist.

- Wirkungen der Transaktionen des Beschwerdeverfahrens

  $Poss(a,s) \Rightarrow (Beschwerde(g,p,r,w,art,do(a,s)) \Leftrightarrow$

  $a=register\_Beschwerde(g,p,r,w,art,t) \vee Beschwerde(g,p,r,w,art,s) \wedge$

  $\neg a=loeschen\_Beschwerde(g,p,r,w,art,t))$

Eine Beschwerde ist im Grundbuchamt registriert, wenn sie gerade registriert wurde, oder sie bereits registriert ist und sie nicht gerade (nach ihrer Bearbeitung) gelöscht wurde.

5.4.4.2 Verfeinerung der Inkonsistenzmarkierungen

Entscheidende Voraussetzung zur Korrektur des Grundbuchs ist die Markierung der Inkonsistenzen. Die Korrektur kann dadurch geschehen, daß inkorrekte Eintragungen gelöscht oder fehlende Eintragungen hinzugefügt werden. Dazu ist es notwendig, die Inkonsistenzmarkierungen zu verfeinern, um diesen Unterschied zwischen inkorrekten und fehlenden Eintragungen abzubilden. Wie dies geschieht, soll das folgende Beispiel verdeutlichen:
Die folgende Formel ist die Definition der Inkonsistenzmarkierung für Eigentumsverhältnisse:

$$Inkons\_Eigentümer(g,p,s) \Leftrightarrow \neg(Eigentümer(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Eigentum,s))$$

Diese Formel kann nach einigen Umformungen in die folgende Form überführt werden:

$$Inkons\_Eigentümer(g,p,s) \Leftrightarrow Eigentümer(g,p,s) \wedge \neg Eingetragenes\_Recht(g,p,Eigentum,s) \vee Eingetragenes\_Recht(g,p,Eigentum,s) \wedge \neg Eigentümer(g,p,s)$$

Diese Formel sagt aus, daß eine Inkonsistenzmarkierung erfolgt, wenn entweder ein Eigentumsrecht materiellrechtlich besteht, aber nicht im Grundbuch eingetragen ist, oder ein Recht im Grundbuch eingetragen ist, das nicht besteht. Dies sind genau die zwei Fälle, die abgebildet werden müssen: im ersten Fall fehlt eine Eintragung im Grundbuch, im zweiten Fall ist eine Eintragung überflüssig.
Fehlende Einträge können nun wie folgt markiert werden:

$$Inkons\_Eigentümer\_ne(g,p,s) \Leftrightarrow Eigentümer(g,p,s) \wedge \neg Eingetragenes\_Recht(g,p,Eigentum,s)$$

Entsprechend erfolgt die Markierung überflüssiger Eintragungen:

$$Inkons\_Eigentümer\_e(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Eigentum,s) \wedge \neg Eigentümer(g,p,s)$$

Analog können die Verfeinerungen für die verbleibenden Markierungen definiert werden:

$$Inkons\_Hypothek\_ne(g,p,w,s) \Leftrightarrow Gläubiger\_Hypothek(g,p,w,s) \wedge \neg Eingetragenes\_Recht(g,p,Hypothek,s) \vee Gläubiger\_Hypothek(g,p,w,s) \wedge \neg Wert\_Recht(g,p,r,w,s)$$

$$Inkons\_Hypothek\_e(g,p,w,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Hypothek,s) \wedge Wert\_Recht(g,p,r,w,s) \wedge \neg Gläubiger\_Hypothek(g,p,w,s)$$

$$Inkons\_Nießbrauch\_e(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Nießbrauch,s) \wedge \neg Nießbraucher(g,p,s)$$

$$Inkons\_Nießbrauch\_ne(g,p,s) \Leftrightarrow Nießbraucher(g,p,s) \wedge \neg Eingetragenes\_Recht(g,p,Nießbrauch,s)$$

Mit Hilfe der verfeinerten Inkonsistenzmarkierungen können nun die korrigierenden Abläufe beschrieben werden.

5.4.4.3 Beschreibung des korrigierenden Ablaufs

Zur Korrektur des Grundbuchs muß geprüft werden, ob eine Beschwerde vorliegt und ob die in ihr bezeichnete Stelle als inkonsistent markiert ist. Wenn das der Fall ist, dann kann in Abhängigkeit der Art der Beschwerde eine Eintragung hinzugefügt oder gelöscht werden.

```
proc gb_korrektur
    (?Beschwerde(g,p,Eigentum,w,loeschen)∧Inkons_Eigentum_e(g,p));
        loeschen_Vollrecht(g,p,Eigentum,now) |
```

(?Beschwerde(g,p,Eigentum,w,eintragen)∧Inkons_Eigentum_ne(g,p));
eintragen_Vollrecht(g,p,Eigentum,now) |
(?Beschwerde(g,p,Hypothek,w,loeschen)∧Inkons_Hypothek_e(g,p,w));
loeschen_Teilrecht_Wert(g,p,Hypothek,w,now) |
(?Beschwerde(g,p,Hypothek,w,eintragen)∧Inkons_Hypothek_ne(g,p,w));
eintragen_Teilrecht_Wert(g,p,Hypothek,w,now) |
(?Beschwerde(g,p,Nießbrauch,w,loeschen)∧Inkons_Nießbrauch_e(g,p));
loeschen_Teilrecht_Nutzung(g,p,Nießbrauch,now) |
(?Beschwerde(g,p,Nießbrauch,w,eintragen)∧Inkons_Nießbrauch_ne(g,p));
eintragen_Teilrecht_Nutzung(g,p,Nießbrauch,now)
endProc

5.5 Zusammenfassung

In diesem Kapitel wurde das formale Modell des Grundbuchsystems erstellt, wie es konzeptionell in Kapitel 3 beschrieben wurde. Folgende Gedanken spielten dabei die zentrale Rolle:

1. Elementare und komplexe Transaktionen beschreiben Veränderungen im Grundbuchsystem.
2. Neben der Modellierung des Grundbuchs ist die Modellierung der materiellen Rechtsverhältnisse notwendig. Auf dieser Basis können Probleme im Zusammenhang mit der Grundbuchunrichtigkeit beschrieben werden, die sich aus dem Verhältnis der beiden Modelle ergeben.

Das Modell bezieht sich nicht auf die vollständige Abbildung des Grundbuchrechts, sondern auf die Beschreibung aussagekräftiger Fälle, die Veränderungen im Grundbuch und der materiellen Welt und Probleme im Zusammenhang mit der Grundbuchunrichtigkeit repräsentieren. Welche das sein können, wurde am Ende dieses Kapitels an einigen Beispielen motiviert.

Es stellt sich die Frage, warum nicht auch für die materiellen Rechtsänderungen Abläufe spezifiziert wurden. Die Antwort ergibt sich aus der Tatsache, daß sich das Handeln von Personen in der materiellen Welt nicht spezifizieren läßt. Man kann es lediglich beobachten. Man kann beobachten, welche materiellen Rechtsänderungen vollzogen, welche Anträge beim Grundbuchamt gestellt und welche Beweisdokumente vorgelegt werden. Wie diese Beobachtungen in das System eingebracht werden und wie sie sich zu den spezifizierten Abläufen verhalten, wird Teil der zu entwickelnden Implementation sein.

Ob das Modell das Grundbuchrecht zufriedenstellend abbildet, ist die zentrale Frage, die an dieser Stelle offen bleibt. Die Überprüfung soll durch Tests mit einem Prototypen durchgeführt werden, dessen Entwicklung noch erfolgen muß. Es ergeben sich also zwei Aufgaben, die verbleiben:

1. Entwicklung eines ausführbaren Prototypen und
2. das Testen des Modells auf Übereinstimmung mit der modellierten Welt.

Dies sind die Probleme und Aufgaben, die im folgenden Kapitel Implementation gelöst werden sollen.

6 Implementation

6.1 Einleitung

Inhalt dieses Kapitels ist die Implementierung einer ausführbaren Spezifikation auf Basis des formalen Modells. Es ergeben sich dabei drei Schwerpunkte:

1. Die formale Begründung der Implementation,
2. die Implementation und
3. der Test der Implementation.

Im Punkt 1 werden die formalen Grundlagen beschrieben, die die Erstellung der Implementation und insbesondere die Korrektheit dieser Implementation bezüglich des formalen Modells begründen. Punkt 2 stellt die Implementation selbst vor. Dabei werden insbesondere die implementationsspezifischen Probleme und ihre Lösung, sowie ihr Zusammenhang mit dem formalen Modell angegeben. Punkt 3 beschäftigt sich mit dem Test der Implementation, also mit der Frage, ob die Implementation und damit das formale Modell korrekt bezüglich des zu modellierenden Teils der Welt sind. Der Test erfolgt anhand eines konkreten Testszenarios, das in der Lage ist, die interessanten Fälle abzubilden. Mit Hilfe dieses Szenarios werden einige dieser interessanten Fälle durchgespielt und dabei überprüft, ob das Programm sich verhält, wie es entsprechend dem konzeptionellen Modell zu erwarten ist.

6.2 Formale Grundlagen der Implementation

Dieser Abschnitt beinhaltet die Realisierung von Punkt 1, die Beschreibung der formalen Grundlagen der Implementation. Die Implementation erfolgt in Prolog (Bratko 1990). Die Korrektheit der Prolog Implementation beruht auf einem Zusammenhang zwischen einer Theorie, die nur aus Definitionen besteht, und der korrekten Implementation einer solchen Theorie in Prolog. Zur Implementation sind zwei Aufgaben zu lösen: Zuerst ist die Spezifikation der elementaren Transaktionen in eine Theorie zu übersetzen, die nur aus Definitionen besteht. Danach müssen diese Definitionen in Prolog Syntax formuliert werden. Auf Basis der Implementation der elementaren Transaktionen erfolgt dann die Definition der GOLOG Prozeduren und die Formulierung eines Interpreters zur Auswertung dieser Prozeduren. Die folgende Darstellung orientiert sich an (Reiter 1998a, S.96ff).

6.2.1 Begriffe und Schreibweisen

Definition 6.1 Definitionen und Definitionstheorien (Reiter 1998a, Definition 4.1.1)

Eine prädikatenlogische Formel erster Ordnung wird Definition genannt, wenn sie die folgende syntaktische Form hat:

$$\forall x_1,\ldots,x_n\ P(x_1,\ldots,x_n) \Leftrightarrow \Phi$$

P ist dabei ein n-stelliges Prädikat (nicht das Gleichheitsprädikat) und Φ *eine prädikatenlogische Formel erster Ordnung mit den freien Variablen* $x_1,\ldots,x_n$.
Eine Definitionstheorie ist eine Theorie, deren Axiome genau eine Definition für jedes Prädikat außer dem Gleichheitsprädikat enthält.
Die Formel $\forall x_1,\ldots,x_n\ P(x_1,\ldots,x_n) \Rightarrow \Phi$ *wird If-Hälfte der Definition von P genannt.*
Ein normales Goal ist eine Konjunktion von Literalen.

Zu beachten ist, daß die Argumente von P unterschiedliche Variablen sein müssen. Also ist $\forall x\,y\ Q(A,f(x),y) \Leftrightarrow \Phi$ keine Definition, unabhängig welche Form Φ hat, da f(x) keine Variable ist.

Die folgenden Begriffe und Schreibweisen sind Grundlagen für Prolog. Eine *Klausel* ist eine Formel der Form $L_1 \wedge ... \wedge L_n \Rightarrow A$. Dabei sind $L_1 ... L_n$ Literale und A eine atomare Formel. Ein Prolog Programm ist eine endliche Sequenz von Klauseln. Prolog benutzt zur Darstellung eine andere Syntax. Die obige Klausel wird in Prolog geschrieben als: `A:-L1,...,Ln`. Anstelle der logischen Negation $\neg$ nutzt man das Symbol *not* („negation as failure") (Clark 1978).

6.2.2 Definitionstheorien und Prolog

Die Grundlage der Implementation ist ein Resultat über den Zusammenhang von Definitionstheorien und zugehörenden Prolog Programmen vom Keith Clark (Clark 1978). Informell gesprochen besagt es, daß aus einer Definitionstheorie ein Prolog Programm gewonnen werden kann, indem man die If-Hälften aller Definitionen als Prolog Klauseln schreibt. Wann immer nun das Prolog Programm ein Goal ableiten kann, ist es aus der zugehörenden Definitionstheorie folgerbar. Scheitert die Ableitung des Goals, dann ist die Negation des Goals aus der Theorie folgerbar.
Um diesen Zusammenhang formal zu beschreiben, ist vorher die Einführung des Begriffs des *geeigneten Prologinterpreters* notwendig:

Definition 6.2 Ein geeigneter Prolog Interpreter (Reiter 1998a, Definition 4.1.3)

Ein geeigneter Prolog Interpreter evaluiert ein Literal der Form not A unter Benutzung von „negation as failure" nur dann, wenn A ein Grundliteral ist. Wenn A kein Grundliteral ist, dann startet der geeignete Prolog Interpreter diesen Beweis nicht.

Wie kann nun aber ein geeigneter Prolog Interpreter erreicht werden? Die Anwort dieser Frage liegt darin, daß der Nutzer, der Prolog Programmierer, dafür verantwortlich ist, daß „negation as failure" nur an Grundatomen vorgenommen wird. Auf dieser Basis kann mit dem folgenden Theorem die Korrektheit der Prolog Implementation begründet werden.

Theorem 6.1 (Clark) (Reiter 1998a, Theorem 4.1.1)

T ist eine Definitionstheorie zusammen mit den folgenden Formeln:
1. Für Paare verschiedener Funktionssymbole f und g gilt:

$$f(x_1,...,x_n) \neq g(y_1,...,y_m)$$

2. Für jedes n-stellige Funktionssymbol gilt:

$$f(x_1,...,x_n) = f(y_1,...,y_n) \Rightarrow x_1 = y_1 \wedge ... \wedge x_n = y_n$$

3. Für jeden Term t[x] anders als x selbst, der die Variable x enthält, gilt:

$$t[x] \neq x$$

Wenn nun ein Prolog Programm aus T dadurch gewonnen werden kann, daß die If-Hälften der Definitionen als Prolog Klauseln geschrieben werden, dann gilt:

1. *Wenn ein geeigneter Prolog Interpreter erfolgreich ein normales Goal G mit der Variablensubstitution Θ ableiten kann, dann gilt $T \models (\forall) G\Theta$. $T \models (\forall) G\Theta$ bezeichnet die Formel G mit der Variablensubstitution Θ, in der alle freien Variablen universal quantifiziert sind.*
2. *Wenn ein geeigneter Prolog Interpreter das normale Goal G nicht ableiten kann, dann gilt $T \models (\forall) \neg G$. $(\forall) \neg G$ bezeichnet die Formel $\neg G$, in der alle freien Variablen universal quantifiziert sind.*

Dieses Theorem garantiert die Korrektheit der Prolog Implementation bezüglich der zugehörenden Definitionstheorie.
Das Theorem beschränkt die Art der damit zu implementierenden Theorien in zweierlei Hinsicht. Einerseits müssen sich die If-Hälften der Definitionen in Prolog Klauseln übersetzen lassen, und andererseits wird die Korrektheit der Implementation nur für normale Goals begründet. Um allgemeinere Theorien implementieren zu können, muß diese

Beschränkung überwunden werden. Dies geschieht durch Angabe von Transformationsregeln, die die If-Hälften von beliebigen Definitionen in eine Form überführen, die sich in Prolog Klauseln übersetzen läßt. Die Einschränkung auf normale Goals läßt sich wie folgt beseitigen: Wenn *G(x)* ein Goal ist, das nicht normal ist, dann wird der Theorie die Definition $Answer(x) \Leftrightarrow G(x)$ hinzugefügt. Als Goal wird nun *Answer(x)* genutzt an Stelle von *G(x)*, was ein normales Goal ist. Das Problem der Beschränkung auf normale Goals wird somit auf das Problem der Übersetzung der If-Hälfte von $Answer(x) \Leftrightarrow G(x)$ in eine Prolog Klausel zurückgeführt.

6.2.3 Die Überführung von Formeln in Klausel Form

Der Ausgangspunkt für die Transformation ist eine Formel der Form W ⇒ A. Sie steht für eine If-Hälfte einer Definition. Das heißt, A ist atomar. W ist eine beliebige prädikatenlogische Formel erster Ordnung. Das Ergebnis der Transformation ist eine in Prolog übersetzbare Formel lt(W)⇒A. lt(W) wird induktiv definiert über die Struktur von W:

Definition 6.3 Transformationsregeln (revised Lloyd-Topor Transformations (Reiter 1998a, S.115))

1. *Wenn W ein Literal ist, dann ist* $lt(W)=W$
2. $lt(W_1 \wedge W_2) = lt(W_1) \wedge lt(W_2)$
3. $lt(W_1 \vee W_2) = lt(W_1) \vee lt(W_2)$
4. $lt(W_1 \Rightarrow W_2) = lt(\neg W_1 \vee W_2)$
5. $lt(W_1 \Leftrightarrow W_2) = lt((W_1 \Rightarrow W_2) \wedge (W_2 \Rightarrow W_1))$
6. $lt(\forall x\ W) = lt(\neg \exists x\ \neg W)$
7. $lt(\neg\neg W) = lt(W)$
8. $lt(\exists x\ W) = lt(W)$
9. $lt(\neg(W_1 \wedge W_2)) = lt(\neg W_1) \vee lt(\neg W_2)$
10. $lt(\neg(W_1 \vee W_2)) = lt(\neg W_1) \wedge lt(\neg W_2)$
11. $lt(\neg(W_1 \Rightarrow W_2)) = lt(\neg(\neg W_1 \vee W_2))$
12. $lt(\neg(W_1 \Leftrightarrow W_2)) = lt(\neg((W_1 \Rightarrow W_2) \wedge (W_2 \Rightarrow W_1)))$
13. $lt(\neg \forall x\ W) = lt(\exists x\ \neg W)$
14. $lt(\neg \exists x\ W) = \neg lt(W)$

Diese Transformationen ermöglichen die Übersetzung einer Definitionstheorie in eine Form, für die Clarks Theorem gilt. Damit ist es möglich, beliebige Definitionstheorien in Prolog zu implementieren.
In der transformierten Formel müssen dann nur noch die Prolog Operatoren eingesetzt werden, das heißt, $\neg \vee \wedge$ sind durch `not ; ,` zu ersetzen.

Beispiel 6.1 Transformierung von Formeln in Prolog Syntax

Als Beispiel dient die Definition der Inkonsistenzmarkierung für inkonsistente Eigentumsverhältnisse:

Inkons_Eigentümer(g,p,s) ⇔ ¬(*Eigentümer(g,p,s)* ⇔
Eingetragenes_Recht(g,p,Eigentum,s))

Es gilt also die folgende Transformation durchzuführen:

lt(¬(Eigentümer(g,p,s) ⇔ *Eingetragenes_Recht(g,p,Eigentum,s)))*

Die Umformung sieht folgendermaßen aus:

lt(¬(Eigentümer(g,p,s) ⇒ *Eingetragenes_Recht(g,p,Eigentum,s)* ∧
Eingetragenes_Recht(g,p,Eigentum,s) ⇒ *Eigentümer(g,p,s)))*
=

$lt(\neg(Eigentümer(g,p,s) \Rightarrow Eingetragenes\_Recht(g,p,Eigentum,s))) \vee$
$lt(\neg(Eingetragenes\_Recht(g,p,Eigentum,s) \Rightarrow Eigentümer(g,p,s)))$
$=$
$lt(\neg(\neg Eigentümer(g,p,s) \vee Eingetragenes\_Recht(g,p,Eigentum,s))) \vee$
$lt(\neg(\neg Eingetragenes\_Recht(g,p,Eigentum,s) \vee Eigentümer(g,p,s)))$
$=$
$lt(\neg\neg Eigentümer(g,p,s)) \wedge lt(\neg Eingetragenes\_Recht(g,p,Eigentum,s)) \vee$
$lt(\neg\neg Eingetragenes\_Recht(g,p,Eigentum,s)) \wedge lt(\neg Eigentümer(g,p,s))$
$=$
$Eigentümer(g,p,s) \wedge \neg Eingetragenes\_Recht(g,p,Eigentum,s) \vee Eingetragenes\_Recht(g,p,Eigentum,s) \wedge \neg Eigentümer(g,p,s)$

Durch einfache Ersetzung läßt sich nun die Prolog Klausel gewinnen:

```
inkons_Eigentuemer(G,P,S) :- eigentuemer(G,P,S), not
eingetragenes_Recht(G,P,eigentum,S);
eingetragenes_Recht(g,P,eigentum,S), not eigentuemer(G,P,S).
```

6.3 Die Implementation elementarer Transaktionen

Im vorigen Abschnitt wurde ein Verfahren vorgestellt, aus einer Definitionstheorie ein Prolog Programm zu gewinnen, das korrekt bezüglich der Theorie ist. Um Spezifikationen von Datenbanktransaktionen in Prolog implementieren zu können, ist es notwendig, diese Spezifikation in eine Definitionstheorie umzuformen. Diese Umformung betrifft die Transaction Precondition Axiome, die Successor State Axiome und die initiale Datenbank.

6.3.1 Transaction Precondition Axiome als Definitionen

Transaction Precondition Axiome haben die Form $Poss(A(x_1,...,x_n),s) \Leftrightarrow \Pi_T(x_1,...,x_n,s)$. Dies ist keine korrekte Definition, da die Argumente von Poss nicht nur aus Variablen bestehen. Für jede Transaktion $A_1,...,A_n$ gibt es ein solches Axiom. Für das Prädikat Poss wird nun eine allgemeine Definition erzeugt.

Definition 6.4 Definition für *Poss* (Reiter 1998a, Definition 4.3.1)

$$Poss(a,s) \equiv (\exists x_1,...\exists x_m(a=A_1(x_1,...,x_m) \wedge \Pi_{A1}(x_1,...,x_m,s))) \vee ... \vee (\exists z_1,...\exists z_m(a=A_n(z_1,...,z_n) \wedge \Pi_{An}(z_1,...,z_m,s))) \quad (6.1)$$

Die If-Hälfte dieser Definition von Poss ist äquivalent zu den folgenden Formeln:

$$\Pi_{A1}(x_1,...,x_m,s) \Rightarrow Poss(A_1(x_1,...,x_m),s),$$

...

$$\Pi_{An}(x_1,...,x_m,s) \Rightarrow Poss(A_n(x_1,...,x_m),s).$$

Diese Formeln können unter Anwendung der Transformationsregeln in Prolog Klauseln überführt werden.

Beispiel 6.2 Übersetzung eines Action Precondition Axioms in Prolog Syntax

$$Poss(register\_Dokument(g,p1,p2,r,t),s) \Leftrightarrow \neg Dokument(g,p1,p2,r,s)$$

In Prolog übersetzt lautet dieses Axiom:

```
poss(register_Dokument(G,P1,P2,R,T),S) :- not
dokument(G,P1,P2,R,S).
```

6.3.2 Successor State Axiome als Definitionen

Unter bestimmten Bedingungen, die die initiale Datenbank erfüllen muß, ist es möglich, die Successor State Axiome für ein Fluent F durch eine Definition für F zu ersetzen. Diese Anforderungen an die initiale Datenbank werden nun beschrieben.

Definition 6.5 Geschlossene initiale Datenbank (Reiter 1998a, Definition 4.3.1)

D sei eine Datenbankspezifikation im Situationskalkül. D_{S0} sei die initiale Datenbank. D_{S0} ist in geschlossener Form, wenn folgende Bedingungen erfüllt sind:

1. *Für jedes Fluent F existiert eine Formel der Form $F(x_1,...,x_m,S_0) \Leftrightarrow \Psi(x_1,...,x_m,S_0)$. Ψ ist eine prädikatenlogische Formel erster Ordnung, deren freie Variablen aus $x_1,...,x_m$ sind und die als einzigen Situationsterm S_0 enthält.*
2. *Für jedes Prädikatensymbol P, das kein Fluent ist, enthält D_{S0} eine Formel $P(x_1,...,x_m) \Leftrightarrow \Theta(x_1,...,x_m)$. Θ ist eine situationsunabhängige prädikatenlogische Formel erster Ordnung, deren freie Variablen aus $x_1,...,x_m$ sind.*
3. *Weiterhin enthält D_{s0} die folgenden Formeln:*
   Für alle Paare von verschiedenen Funktionssymbolen der Sorte Objects f und g gilt: $f(x_1,...,x_m) \neq g(y_1,...,y_n)$.
   Für Funktionssymbole der Sorte Objects gilt: $f(x_1,...,x_n)=f(y_1,...,y_n) \Rightarrow x_1=y_1 \wedge ... \wedge x_n=y_n$.
   Für jeden Term t der Sorte Objects ungleich der Variable x der Sorte Objects gilt: $t[x] \neq x$.
   Für jeden Term a der Sorte Transactions ungleich der Variable x der Sorte Transactions gilt: $a[x] \neq x$.

Bei den unter 3 genannten Formeln handelt es sich um die von Clarks Theorem geforderten Unique Names Axiome.
Die Beschränkung auf geschlossene initiale Datenbanken eröffnet die Möglichkeit, Definitionen für die Fluents zu formulieren.

Definition 6.6 Definitionen für Fluents (Reiter 1998a, Definition 4.3.3)

F ist ein Fluent, dessen Successor State Axiom $F(x_1,...,x_n do(a,s)) \Leftrightarrow \Phi_F(x_1,...,x_n,a,s)$ lautet. D_{S0} ist eine geschlossene initiale Datenbank und enthält folglich die Formel $F(x_1,...,x_m,S_0) \Leftrightarrow \Psi_F(x_1,...,x_m,S_0)$. Dann wird die folgende Formel die Definition für F genannt:

$$F(x_1,...,x_n,s) \Leftrightarrow s=S_0 \wedge \Psi_F(x_1,...,x_m,S_0) \vee \exists a \exists s' \; s=do(a,s') \wedge \Phi_F(x_1,...,x_n,a,s') \qquad (6.2)$$

Die If-Hälfte dieser Formel ist äquivalent zu den folgenden zwei Formeln:

$$\Psi_F(x_1,...,x_m,S_0) \Rightarrow F(x_1,...,x_m,S_0)$$

$$\Phi_F(x_1,...,x_n,a,s') \Rightarrow F(x_1,...,x_n,do(a,s))$$

Durch Anwendung der Transformationsregeln auf Ψ_F und Φ_F können diese Formeln in Prolog Klauseln überführt werden.

Beispiel 6.3 Überführung eines Successor State Axioms in Prolog Syntax

$$Poss(a,s) \Rightarrow (Dokument(g,p1,p2,r,do(a,s)) \Leftrightarrow$$
$$a=register\_Dokument(g,p1,p2,r,t) \vee Dokument(g,p1,p2,r,s))$$

In Prolog übersetzt lautet diese Formel:

```
dokument(G,P1,P2,R,do(A,S)):- A=register_Dokument(G,P1,P2,R,T);
dokument(G,P1,P2,R,S).
```

6.3.3 Das Implementationstheorem

Nach der Formulierung der Definitionen für die Updatespezifikation ist es nun notwendig, den Zusammenhang zwischen der ursprünglichen Spezifikation und den gewonnenen Definitionen zu formulieren. Der entscheidende Punkt ist hierbei die Frage, ob die Menge der Definitionen mit der Updatespezifikation äquivalent ist und somit überhaupt als Grundlage für die Implementation dienen kann. Diese Frage wird durch das folgende Theorem geklärt.

Theorem 6.2 (Reiter 1998a, Theorem 4.3.1)

Sei D eine Datenbank (Definition 4.3) im Situationskalkül, deren initiale Datenbank sich in geschlossener Form befindet. D_{S0}^{-} wird aus D_{S0} gewonnen durch Weglassen aller Äquivalenzen für relationale Fluents. Folglich enthält D_{S0}^{-} die Definitionen der situationsunabhängigen Prädikate und Unique Names Axiome.

D_{tp}^{-} ist die Definition für Poss (6.1).
D_{ss}^{-} ist die Menge der Definitionen für Fluents (6.2).
D_{unsit} enthält die folgenden Unique Names Axiome für Situationen aus Σ:

$$S_0 \neq do(a,s)$$

$$do(a_1,s_1)=do(a_2,s_2) \Rightarrow a_1=a_2 \wedge s_1=s_2$$

sowie die Instanzen des Schemas

$$t[s]\neq s$$

für jeden Term t der Sorte situation ungleich s, der die Variable s enthält.
Dann sind D und $D_{S0}^{-} \cup D_{tp}^{-} \cup D_{ss}^{-} \cup D_{unt} \cup D_{unist}$ für Formeln, für die Regression anwendbar ist, äquivalent im folgenden Sinn:
Wenn G eine Formel ist, für die Regression anwendbar ist, dann gilt:

$$D \models G \Leftrightarrow D_{S0}^{-} \cup D_{tp}^{-} \cup D_{ss}^{-} \cup D_{unt} \cup D_{unist} \models G$$

Bezüglich des Schemas $t[s]\neq s$ ist zu sagen, daß alle seine Instanzen bereits aus den Unique Names Axiomen für Situationen abgeleitet werden können. Es wird deshalb im folgenden weggelassen.
Das obige Theorem drückt aus, daß sich zu einer Datenbankspezifikation eine äquivalente Definitionstheorie bilden läßt (äquivalent bezüglich der Beweisbarkeit von Formeln, für die Regression anwendbar ist). Diese Theorie $D_{S0}^{-} \cup D_{tp}^{-} \cup D_{ss}^{-} \cup D_{unt} \cup D_{unist}$ ist nun wiederum eine Definitionstheorie, für die Clarks Theorem gilt. Es läßt sich also dazu eine korrekte Prolog Implementation finden, wie das folgende Implementationstheorem ausdrückt:

Korollar 6.1 Implementationstheorem (Reiter 1998a, Corollary 4.3.1)

Unter den Bedingungen von Theorem 6.2 bilden die folgenden Formeln nach Anwendung der Transformationsregeln eine korrekte Prolog Implementation einer Datenbank im Situationskalkül D:

1. *Für jede Definition eines situationsunabhängigen Prädikats P in D_{S0} der Form $P(x_1,...,x_m) \Leftrightarrow \Theta(x_1,...,x_m)$ die Formel $\Theta(x_1,...,x_m) \Rightarrow P(x_1,...,x_m)$,*
2. *für jede Formel in D_{S0} der Form $F(x_1,...,x_m,S_0) \Leftrightarrow \Psi_F(x_1,...,x_m,S_0)$ die Formel $\Psi_F(x_1,...,x_m,S_0) \Rightarrow F(x_1,...,x_m,S_0)$,*
3. *für jedes Transaction Precondition Axiom der Form $Poss(A(x_1,...,x_n),s) \Leftrightarrow \Pi_A(x_1,...,x_n,s)$ die Formel $\Pi_A(x_1,...,x_n,s) \Rightarrow Poss(A(x_1,...,x_n),s)$,*
4. *für jedes Successor State Axiom der Form $Poss(a,s) \Rightarrow (F(x_1,...,x_n do(a,s)) \Leftrightarrow \Phi_F(x_1,...,x_n,a,s))$ die Formel $\Phi_F(x_1,...,x_n,a,s') \Rightarrow F(x_1,...,x_n,do(a,s))$.*

Für die Formulierung einer Anfrage, die eine Formel sein muß, für die Regression anwendbar ist, müssen zuerst die Transformationsregeln angewandt und dann die Anfrage gestellt werden.

Das Implementationstheorem schafft die Voraussetzung, Spezifikationen von Transaktionen in Prolog zu implementieren.

6.3.4 Regression und Prolog

Regression ist das Werkzeug, um in einer Spezifikation im Situationskalkül Formeln zu beweisen. Folglich ist es sinnvoll, die Menge der Formeln auf die zu beschränken, für die Regression anwendbar ist. Ist diese Einschränkung getan, so besagt Theorem 6.2, daß sich dazu eine äquivalente Definitionstheorie bilden läßt, für die Clarks Theorem gilt. Für diese Theorie läßt sich eine korrekte Implementation angeben. Korrekt ist sie bezüglich der Regression. Wann immer eine Formel durch Regression ableitbar ist, ist auch der Prolog Interpreter erfolgreich. Das heißt wiederum, daß der Prolog Interpreter als Implementation des Regressionsoperators angesehen werden kann. Für den Zweck der Implementation bedeutet dies, daß kein eigener Regressionsbeweiser erstellt werden muß und Prolog von sich aus die notwendigen Fähigkeiten mitbringt. Folglich ist an dieser Stelle kein weiterer Aufwand erforderlich.

6.4 Die Implementation komplexer Transaktionen

Der vorige Abschnitt begründete die Implementation elementarer Transaktionen in Prolog. Auf dieser Basis werden nun komplexe Transaktionen definiert. Wichtig ist dabei die Frage, wie die komplexen Transaktionsausdrücke, also die GOLOG Prozeduren, ausgewertet werden. Es muß also ein Interpreter für GOLOG Programme erstellt werden. Ein GOLOG Interpreter wurde an der Universität von Toronto entwickelt (COROBO 1998). Er wird erweitert um die Fähigkeiten des sequentiellen Temporalen Situationskalküls und RGOLOGs, die in Kapitel 4 beschrieben wurden.
Als Prolog Implementation wird ECLiPSe (IC-Parc 1996-1998) genutzt.

6.4.1 Der GOLOG Interpreter

Der Implementation liegen folgende Gedanken zu Grunde. Die Auswertung der GOLOG Programme basiert auf der Spezifikation der elementaren Transaktionen. Diese lassen sich, wie im vorigen Abschnitt gezeigt, in Prolog Syntax übersetzen. In Kapitel 4 wurde gezeigt, daß die Auswertung der GOLOG Prozeduren sich auf den Beweis von Formeln zurückführen läßt, für die Regression anwendbar ist.
Das Implementationstheorem garantiert die Korrektheit der Prolog Implementation für den Zweck des Beweises von Formeln, für die Regression anwendbar ist. Folglich kann eine korrekte Prolog Implementation für GOLOG Programme erstellt werden.
Entsprechend der Definition der komplexen Transaktionen (Abschnitt 4.6.2) wird der Interpreter für die komplexen Transaktionen erstellt. Der folgende Code implementiert das Makro *Do*. Für die vollständige Implementation wird auf Anhang A verwiesen.

```
do(E1 : E2,S,S1) :- do(E1,S,S2), do(E2,S2,S1).
do(?(P),S,S) :- holds(P,S).
do(E1 # E2,S,S1) :- do(E1,S,S1) ; do(E2,S,S1).
do(if(P,E1,E2),S,S1) :- do((?(P) : E1) # (?(-P) : E2),S,S1).
do(star(E),S,S1) :- S1 = S ; do(E : star(E),S,S1).
do(while(P,E),S,S1):- do(star(?(P) : E) : ?(-P),S,S1).
do(pi(V,E),S,S1) :- sub(V,_,E,E1), do(E1,S,S1).
do(E,S,S1) :- proc(E,E1), do(E1,S,S1).
do(E,S,do(E,S)) :- primitive_action(E), poss(E,S).
```

Das `do(E,S,S1)` Prädikat hat drei Argumente: einen komplexen Transaktionsausdruck E, einen Term S, der die initiale Situation bezeichnet, und den Term S1, der die finale Situation bezeichnet. Solch ein Prädikat wird als Anfrage an das System gestellt und liefert als Antwort die Variablenbindung für die finale Situation S1 bei die Auswertung des komplexen Transaktionsausdrucks E. Diese Variablenbindung kann dann als Programmlauf betrachtet werden, liefert sie doch eine Kette von Transaktionen, die die Situationsübergänge verursachten.
Eine komplexe Transaktion kann die folgende Form haben:

- *E1 : E2*, die Sequenz oder Nacheinanderausführung zweier Transaktionen.

- *?(P)*, P ist eine Bedingung, die überprüft wird.
- *E1 # E2*, die nichtdeterministische Auswahl zweier Transaktionen.
- *if(P,E1,E2)*, wenn P gilt, dann führe die Transaktion E1 aus, sonst E2.
- *star(E)*, die nichtdeterministische Wiederholung der Transaktion E.
- *while(P,E)*, solange P gilt, führe E aus.
- *Pi(V,E)*, die nichtdeterministische Zuweisung einer Variablen V in einer Transaktion E.
- *E*, wenn E der Name einer nutzerdefinierten elementaren Transaktion oder ein Prozeduraufruf ist.

Eine Bedingung P kann die Form `p1 & p2, p1 v p2, -p, p1 <=> p2, all(k,p)` oder `some(k,p)` haben. k ist hierbei eine Prolog Konstante und p eine Bedingung, die k benutzt. Für die Auswertung einer solchen Bedingung wurden die Transformationsregeln für Bedingungen implementiert. Für diese Implementation wird auf Anhang A verwiesen. Zur Auswertung einer Negation (-p) wird der „negation as failure" Mechanismus benutzt. Zu bemerken ist an dieser Stelle, daß die Transformationsregeln nur für Bedingungen implementiert wurden. Für die Spezifikation der elementaren Transaktionen liegt diese Arbeit in des Nutzers Hand.
Zur Definition von Prozeduren existieren Klauseln der Form *proc(name,körper)*. Dabei ist *name* der Name der Prozedur und *körper* die Implementation der Prozedur, also der Prozedurkörper. Elementare Transaktionen E werden durch eine Klausel der Form *primitive_action(E)* gekennzeichnet.
GOLOG Prozeduren werden ohne die Angabe der Situationsargumente für Transaktionen formuliert. Zur Auswertung dieser Prozeduren ist die Wiederherstellung dieser Situationsargumente notwendig. Dazu wird für jedes Fluent mit den Argumenten X eine Klausel formuliert, die diese Wiederherstellung realisiert: *restoreSitArg(Fluent(X),S,Fluent(X,S))*.
Die Übersetzung von GOLOG Programmen in den Situationskalkül kann zu Formeln zweiter Ordnung führen. Wie kann nun ein Prolog Interpreter für eine solche Formel eine korrekte Implementation sein? Als Argument soll hier das Beispiel der nichtdeterministischen Wiederholung von Transaktionen dienen, also für einen Transaktionsausdruck a der Ausdruck $Do(a^*,s,s')$. Es gilt aber die folgende Äquivalenz: $Do(a^*,s,s') \Leftrightarrow s=s' \vee Do(a;a^*,s,s')$. Das bedeutet, daß dieser Ausdruck korrekt ausgewertet werden kann, ohne die Formel zweiter Ordnung zu gebrauchen. Dies genau geschieht bei der Implementation der nichtdeterministischen Wiederholung im oben beschriebenen GOLOG Interpreter.

6.4.2 Der sequentielle temporale GOLOG Interpreter

Der sequentielle temporale Situationskalkül erweitert den Situationskalkül um einen quantitativen Zeitbegriff. Dies wird dadurch erreicht, daß Transaktionen ein Argument erhalten, das den Zeitpunkt des Auftretens der Transaktion angibt. Die Funktion *time* gibt den Zeitpunkt einer Transaktion an und die Funktion *start* den Beginn einer Situation. An den Basisaxiomen des Situationskalküls änderte sich nur die Definition der < Relation für Situationen, damit keine rückwärts laufende Zeit möglich ist (Axiom (4.16)). Dies kann nun einfach umgesetzt werden, indem die Makrodefinition von *Do* angepaßt wird. Dabei wird Formel (4.13) ersetzt durch:

$$Do(a,s,s') \stackrel{def}{=} Poss(a[s],s) \wedge start(s) \leq time(a) \wedge s' = do(\alpha[s],s)$$

Implementiert wird diese Definition im GOLOG Interpreter einfach dadurch, daß die Klausel

```
do(E,S,do(E,S)) :- primitive_action(E), poss(E,S).
```

durch

```
do(E,S,do(E,S)) :- primitive_action(E), poss(E,S), start(S,T1), time(E,T2),
T1$<=T2.
```

ersetzt wird.
Weiterhin müssen wegen der Einführung des Prädikats *start* die folgenden Klauseln hinzugefügt werden:

```
restoreSitArg(start(T),S,start(S,T)).
start(do(A,S),T) :- time(A,T).
```

Erinnert werden soll, daß für jede Transaktion A ein Atom der Form *time(A(T),T)* der initialen Datenbank hinzugefügt wird.

6.4.3 Der RGOLOG Interpreter

Der Interpreter von RGOLOG implementiert die Auswertung des Makros DoR aus Abschnitt 4.6.4 (Formel(4.18)). Ziel war es, ein GOLOG zu definieren, das die Fähigkeit hat, auf mögliche externe Transaktionen oder bei Eintritt bestimmter Bedingungen zu reagieren. Dazu mußte einerseits die Möglichkeit geschaffen werden, daß externe Transaktionen auftreten, und andererseits das Verfahren definiert werden, das die Reaktionen des GOLOG Programms festlegt. Die Idee war es, daß nach jeder Ausführung einer Transaktion des laufenden GOLOG Programms ein weiteres Programm *rules* ausgeführt wird, das einerseits die externen Transaktionen aufnimmt und andererseits die Reaktionen des Systems auf eventuelle Veränderungen auslöst. Diese Gedanken führen zu der folgenden Implementation (Reiter 1998a, S.209):

```
doR(E1 : E2,Rules,S,S1) :- doR(E1,Rules,S,S2), doR(E2,Rules,S2,S1).
doR(?(P),Rules,S,S) :- holds(P,S).
doR(E1 # E2,Rules,S,S1) :- doR(E1,Rules,S,S1) ; doR(E2,Rules,S,S1).
doR(if(P,E1,E2),Rules,S,S1) :- doR(?(P) : E1 # ?(-P) : E2,Rules,S,S1).
doR(star(E),Rules,S,S1) :- S1 = S ; doR(E : star(E),Rules,S,S1).
doR(while(P,E),Rules,S,S1):- doR(star(?(P) : E) : ?(-P),Rules,S,S1).
doR(pi(V,E),Rules,S,S1) :- sub(V,_,E,E1), doR(E1,Rules,S,S1).
doR(E,Rules,S,S1) :- proc(E,E1), doR(E1,Rules,S,S1).
doR(E,Rules,S,S1) :- primitive_action(E), poss(E,S),
doR(Rules,Rules,do(E,S),S1).
```

Von dem obigen GOLOG Interpreter unterscheidet es sich dadurch, daß das Makro doR einen weiteren Parameter erhält, eben genau das Programm *rules*. Die Auswertung der Transaktionen ändert sich dadurch nur für den Fall, daß der übergebene Transaktionsausdruck eine elementare Transaktion ist. Dann wird nach Ausführung der elementaren Transaktion das Programm *rules* zur Ausführung gebracht. Für alle komplexen Transaktionsausdrücke wird der Parameter *rules* lediglich weitergereicht.

6.4.4 Der GOLOG Interpreter zur Implementation eines Grundbuchsystems

Nach der Einführung der GOLOG Versionen verbleibt eine Aufgabe, die Verbindung des Interpreters für sequentielles temporales GOLOG und des Interpreters für RGOLOG herzustellen. Anders gesagt: die Erweiterung von RGOLOG um einen quantitativen Zeitbegriffs durchzuführen. Hierzu werden die Gedanken aus den beiden vorigen Abschnitten einfach zusammengefaßt.
Die Klausel

```
doR(E,Rules,S,S1) :- primitive_action(E), poss(E,S),
doR(Rules,Rules,do(E,S),S1).
```

wird ersetzt durch

```
doR(E,Rules,S,S1) :- primitive_action(E), poss(E,S),start(S,T1), time(E,T2),
T1$<=T2, doR(Rules,Rules,do(E,S),S1).
```

Weiterhin werden die oben beschriebenen Klauseln für *start* hinzugefügt.
Dieser Interpreter wird im weiteren als reaktiver sequentieller temporaler GOLOG Interpreter bezeichnet. Er verfügt nun über die Fähigkeiten, die für die Implementation des Grundbuchsystems notwendig sind. Dies wird im folgenden Abschnitt geschehen.

6.5 Die Implementation des Grundbuchsystems

Dieser Abschnitt ist nicht der vollständigen Darstellung des implementierten Codes gewidmet. Vielmehr geht es darum, die Konzepte vorzustellen, die dieser Implementation zugrunde liegen. Es wird gezeigt, wie das formale Modell umgesetzt wurde. Diese Darstellung erfolgt anhand von Beispielen. Der vollständige Code ist in Anhang B aufgelistet.

6.5.1 Grundgedanken

Das Modell des Grundbuchsystems besteht aus zwei wichtigen Teilen, der Repräsentation des Grundbuchs und der Repräsentation der materiellen Rechtsverhältnisse. Veränderungen in den materiellen Rechtsverhältnissen sollen als Beobachtungen in der materiellen Welt betrachtet werden. Die Idee, diesen Gedanken umzusetzen, ist die folgende: Beobachtungen werden als externe Transaktionen modelliert und werden dem System somit von außen eingegeben. Die Verbindung von formellem und materiellem Recht bilden Beweisdokumente und Anträge. Sie stellen den Nachweis der Veränderungen der materiellen Rechtsverhältnisse für das Grundbuch dar und somit die Grundlage für ihre Abbildung im Grundbuch. Bei der Erstellung des Modells des Grundbuchs wurden die Transaktionen, die das formelle Recht betreffen, ebenfalls als externe Transaktionen bezeichnet, um darauf hinzuweisen, daß sie von für das Grundbuch externen Ereignissen ausgelöst werden. Menschen stellen Anträge beim Grundbuchamt, Menschen legen Beweisdokumente vor. Man kann sie in diesem Sinn auch als Beobachtungen über Ereignisse in der materiellen Welt auffassen, wie die Veränderungen in den materiellen Rechtsverhältnissen selbst.
Diese Implementation geht basierend auf diesen Überlegungen folgenden Weg: Grundlage für die Implementation ist der reaktive sequentielle temporale GOLOG Interpreter. Elementare Transaktionen werden spezifiziert und bilden die Grundlage für die Definition komplexer Transaktionen des Grundbuchs. Komplexe Transaktionen bilden die Grundbuchverfahren ab. Externe Transaktionen bilden keine komplexen Abläufe, würde das doch bedeuten, daß man ihr auftreten und ihre Struktur spezifizieren könnte. Beobachtungen der Abläufe in der materiellen Welt betreffen Veränderungen in den materiellen Rechtsverhältnissen oder die Handlungen von Menschen, diese Veränderungen im Grundbuch abbilden zu lassen, also zum Beispiel Registrieren von Dokumenten. Beide Arten von Veränderungen werden bezüglich des GOLOG Interpreters als externe Transaktionen implementiert.

6.5.2 Implementation des Zeitbegriffs

Wichtig für die Implementation ist die Integration des Zeitbegriffs. Zeitabläufe werden vom Stattfinden der Ereignisse in der materiellen Welt bestimmt. Diese Ereignisse bestimmen auch die Zeitpunkte der Ereignisse im Grundbuch. Die Transaktionszeiten der externen Transaktionen legen also die Zeiten der internen Transaktionen fest. Für den Zweck dieser Implementation wird die idealisierende Annahme getroffen, daß interne Transaktionen keine Zeit verbrauchen, so daß die Zeit ausschließlich vom Auftreten externer Transaktionen bestimmt wird. Diese Grundgedanken werden in der Implementation wie folgt umgesetzt:

- Definition des Prädikats *now(S,T):*

```
now(S,T) :- start(S,T).
```

```
restoreSitArg(now(T),S,now(T,S)).
```

Das Prädikat definiert den aktuellen Zeitpunkt als Synonym für den Startzeitpunkt der aktuellen Situation.

- Der Aufruf einer internen Transaktion:
  Wenn nun eine interne Transaktion innerhalb einer GOLOG Prozedur aufgerufen wird, zum Beispiel die Transaktion *start_Grundbuchverfahren(G,T),* so erfolgt die Angabe der Transaktionszeit wie folgt:

```
pi(t, (?(now(t)) : start_Grundbuchverfahren(G,t))
```

  Das Prädikat *restoreSitArg* ordnet dem Prädikat *now(t)* die aktuelle Situation zu. Die Transaktion *start_Grundbuchverfahren* erhält als Transaktionszeit die Zeit der aktuellen Situation t, die durch *now(t)* bestimmt ist.
- Das Stattfinden einer externen Transaktion:
  Wenn eine externe Transaktion stattfindet, zum Beispiel *register_Dokument (g1,p1,p2,Eigentum,20),* so legt dies die aktuelle Situationszeit auf 20 fest. Für jede weitere interne Transaktion würde das Prädikat *now(t)* einen Wert von 20 für t festlegen.

Die beschriebene Vorgehensweise setzt den Gedanken um, daß die Zeit im System ausschließlich vom Auftreten externer Transaktionen bestimmt wird. Die zwischen externen Transaktionen ablaufenden internen Transaktionen finden insofern gleichzeitig statt, daß sie zwar in ihrer Abfolge sequentiell geordnet sind, aber die gleichen Transaktionszeiten haben. Auch wenn dies in dieser Implementation nicht getan wird, kann auch den internen Transaktionen eine echte Zeitdauer zugewiesen werden, indem von dem oben beschriebenen Schema abgewichen wird. Dies ist auch für eine reale Anwendung notwendig, da die Annahme, daß die interne Transaktionen keine Zeit verbrauchen, eine Vereinfachung darstellt, die in der Wirklichkeit nicht anwendbar ist.

6.5.3 Die Visualisierung der Daten

Für den Test der Implementation ist es unumgänglich herauszufinden, was im System gilt. Hierzu sind zwei Aspekte wichtig:

1. Welche Transaktionen haben stattgefunden?
2. Welche Wahrheitswerte haben die Fluents nach dem Stattfinden dieser Transaktionen?

Es ist also die Aufgabe, die Transaktionsliste auszugeben und außerdem alle Variablenbindungen für Fluents zu finden, also die Tupel, die Inhalt der Datenbank sind.

- Die Ausgabe der Transaktionsliste:

```
prettyPrintSituation(S) :- makeActionList(S,Alist), nl,
       write(Alist), nl.
makeActionList(s0,[]).
makeActionList(do(A,S),L) :- makeActionList(S,L1), append(L1,[A],L).
```

  Es wird rekursiv eine Liste der die übergebende Situation bestimmenden Transaktionen erstellt und auf dem Bildschirm ausgegeben.
- Die Ausgabe der Wahrheitswerte der Fluents:
  Diese Aufgabe ist auf das Beantworten von Anfragen (Abschnitt 4.7) zurückzuführen. Dieses Problem ist mit Hilfe des Regressionsoperators gelöst worden. Wie bereits in diesem Kapitel (6.3.4) begründet, wird die Funktion des Regressionsoperators durch den Unifikationsmechanismus von Prolog übernommen und folglich sind keine weiteren Aufwendungen notwendig. Die Visualisierung der Daten erfolgt dann auch unter Nutzung des in Prolog integrierten Prädikats *findall* (Bratko 1990, S.182ff.). Es erstellt eine Liste aller Objekte, die ein Goal erfüllen.

```
makeGrundstuecksList(List,S) :- findall (G,grundstueck(G,S),List).
printGrundstuecksList(S) :- write ("Grundstuecke: "),
       makeGrundstuecksList(List,S), write(List), nl.
```

Diese Beispiel baut die Liste der Objekte G auf, die in der Situation S das Goal *grundstueck(G,S)* erfüllen.

6.5.4 Das Hauptprogramm

Das formale Modell spezifizierte lediglich die elementaren und komplexen Transaktionen. Um eine ausführbare Spezifikation zu erhalten, ist es notwendig, diese Spezifikation in ein Programm zu integrieren, das das formale Modell und die implementationsspezifischen Elemente (z.B. Visualisierung der Daten) verbindet.

```
gamt :- doR(gba,rules,s0,S), chooseTimes(S),
prettyPrintSituation(S),printFluentList(S).
```

Das Hauptprogramm heißt *gamt*. Es führt die Prozedur *gba* aus, so daß nach jeder Ausführung einer elementaren Transaktion aus *gba* das Programm *rules* ausgeführt wird.
Prinzipiell liefert die Abarbeitung von *doR(gba,rules,s0,S)* keine vollinstanzierte Liste von Situationen S. Dies liegt daran, daß die Ausführungszeiten der Transaktionen in dieser Sequenz nicht eindeutig bestimmt ist. Es besteht eine Vielzahl von Lösungen. Es muß also eine dieser Lösungen ausgewählt werden. Dies tut das Prädikat *chooseTimes(S)* (Reiter 1998a, S.178).

```
chooseTimes(s0).
chooseTimes(do(A,S)) :- chooseTimes(S), time(A,T), rmin(T).
```

Hierbei wird durch *rmin(T)* (IC-Parc 1998) eine minimale Lösung für T ausgewählt. Durch *chooseTimes(S)* erfolgt dies rekursiv für jede Transaktion in S.
Die Prädikate *prettyPrintSituation(S)* und *printFluentList(S)* geben die Liste der erfolgten Transaktionen und die Werte der Fluents nach diesen Transaktionen auf dem Bildschirm aus.
Um die Verbindung zwischen dem Programm und der Implementation des formalen Modells herzustellen, muß die GOLOG Prozedur *gba* genauer beschrieben werden:

```
proc(gba,
    while (-(halt),
       (if(-(some(an, some(g1, some(g2, some(fl, some(p1, some(p2,
            some(w,antrag(an,g1,g2,fl,p1,p2,w)))))))),
       % then
             pi(t, ?(now(t)) : wait(t)),
       % else
            (pi (g1, pi(g2, pi(fl, pi(p1, pi(p2, pi(w,
              (?(antrag(transfer_e,g1,g2,fl,p1,p2,w)):
                 gb_transfer_Eigentum(g1,p1,p2 ):
                 pi(t, ?(now(t)):
                     loesche_Antrag(transfer_e,g1,g2,fl,p1,p2,w,t)))#
              (?(antrag(hypothek_e,g1,g2,fl,p1,p2,w)) :
                 gb_eintragen_Hypothek(g1,p1,w):
                 pi(t, ?(now(t)):
                     loesche_Antrag(hypothek_e,g1,g2,fl,p1,p2,w,t)))#
              (?(antrag(hypothek_l,g1,g2,fl,p1,p2,w)):
                 gb_loeschen_Hypothek(g1,p1,w):
                 pi(t, ?(now(t)):
                     loesche_Antrag(hypothek_l,g1,g2,fl,p1,p2,w,t)))#
              (?(antrag(nieszbrauch_e,g1,g2,fl,p1,p2,w)):
                 gb_eintragen_Nieszbrauch (g1,p1):
                 pi(t, (now(t)):
                     loesche_Antrag(nieszbrauch_e,g1,g2,fl,p1,p2,w,t))) #
              (?(antrag(teilung_e,g1,g2,fl,p1,p2,w)):
                 gb_Teilung (g1,g2,fl,p1):
                 pi(t, ?(now(t)):
                     loesche_Antrag(teilung_e,g1,g2,fl,p1,p2,w,t))) #
              (?(antrag(vereinigung_e,g1,g2,fl,p1,p2,w)):
                 gb_Vereinigung (g1,g2,p1):
                 pi(t, ?(now(t)):
                     loesche_Antrag(vereinigung_e,g1,g2,fl,p1,p2,w,t))
              )))))))))
        )
).
```

Die Prozedur prüft, ob Anträge auf Eintragung in das Grundbuch vorliegen. Wenn ja, dann wird die entsprechende Prozedur zur Erreichung dieser Eintragung aufgerufen. Danach wird

der Antrag gelöscht. Die Prozeduren, die hier aufgerufen werden (z.B. *gb_transfer_Eigentum*), stellen die Implementation der Spezifikation der komplexen Abläufe dar, wie sie im formalen Modell beschrieben wurden. Sie unterscheiden sich nur an einer Stelle. Die Prüfung, ob der notwendige Antrag vorliegt, geschieht außerhalb dieser Prozeduren. Dies ist begründet durch die dadurch vereinfachte Implementation.
Liegt kein Antrag vor, dann wird die Transaktion *wait* aufgerufen. Sie hat keinerlei Wirkung auf die Fluents. Die Bedeutung der *wait* Transaktion liegt darin, daß nach ihrer Ausführung wieder eine externe Transaktion möglich ist.
Das Programm läuft solange, bis das Prädikat *halt* gilt. Dies geschieht, wenn die Transaktion *stop* aufgerufen wird. Das heißt, daß das Programm erst dann terminiert, wenn der Aufruf dieser Transaktion erfolgt.

6.5.5 Die Implementation der materiellen Rechtsverhältnisse

Materielle Rechtsverhältnisse betreffen die Objekte des materielle Rechts. Veränderungen in diesen materiellen Rechtsverhältnissen ergeben sich durch Veränderungen der Objekte oder der Rechtsverhältnisse selbst. Veränderungen werden abgebildet von Transaktionen, die das Handeln von Personen in der Welt repräsentieren. Wichtig ist die Tatsache, daß Veränderungen in den materiellen Rechtsverhältnissen als elementare Transaktionen abgebildet werden, da die sie verursachenden Abläufe nicht vollständig spezifizierbar sind. Ein weiterer zentraler Gedanke ist die Tatsache, daß die Vorbedingungen für das Stattfinden von materiellrechtlichen Veränderungen das Vorliegen der materiellen Voraussetzungen ist. Als Beispiel für die Implementation der materiellen Rechtsverhältnisse soll im folgenden der Verkauf von Grundstücken und dessen Wirkung auf die Eigentumsverhältnisse dienen.
Zu bemerken ist noch, daß sich die formale Beschreibung des Modells, die hier implementiert werden soll, bereits in einer Form befindet, die in Prolog Syntax übersetzbar ist. Folglich müssen keine Transformationsregeln angewandt werden.

```
poss(verkaufe_Grundstueck(G,P1,P2,T),S) :- grundstueck(G,S), person(P1,S),
       person(P2,S), eigentuemer(G,P1,S).

eigentuemer(G,P,do(A,S)) :-
       A=verkaufe_Grundstueck(G,_,P,T);
       A=teile_Grundstueck(_,G,P,_,T);
       eigentuemer(G,P,S),
       not A=verkaufe_Grundstueck(G,P,_,T),
       not A=vereinige_Grundstuecke(_,G,P,T).
```

Die Vorbedingung legt die materiellrechtlichen Voraussetzung fest. Das ist einerseits die Tatsache, daß die übergebenen Bezeichner auch wirklich Objekte des materiellen Rechtsverkehrs (Grundstücke, Personen) bezeichnen, und andererseits die Tatsache, daß die Person, die ein Grundstück verkaufen will, auch materiellrechtlich dessen Eigentümer ist.
Die Eigentumsverhältnisse an einem Grundstück drückt das Prädikat `eigentuemer` aus. Sie können sich ändern durch Verkauf eines Grundstücks, durch Grundstücksteilung oder Grundstücksvereinigung. Eine Person wird Eigentümer eines Grundstücks, wenn sie das Grundstück kauft oder dieses Grundstück durch Teilung im eigenen Besitz geschaffen hat. Eine Person verliert das Eigentumsrecht, wenn sie es verkauft oder dieses Grundstück durch Vereinigung mit einem anderen Grundstück aufhört zu bestehen.

6.5.6 Die Implementation des Grundbuchs

Veränderungen der materiellen Rechtsverhältnisse werden im Grundbuch dadurch abgebildet, daß Eintragungen erfolgen oder gelöscht werden. Im Gegensatz zu den Veränderungen in der materiellen Welt lassen sich die Abläufe im Grundbuchamt spezifizieren, sind sie doch durch Rechtsvorschriften geregelt. Teil der Implementation können also Abläufe sein. Ein weiterer wichtiger Unterschied ergibt sich durch die Tatsache, daß das Grundbuchamt nicht die materiellrechtlichen Voraussetzungen prüft, sondern das Vorhandensein von Dokumenten, die die materiellrechtlichen Veränderungen beweisen.

Analog dem obigen Beispiel soll die Implementation von Veränderungen in den Eigentumsrechten an Grundstücken angegeben werden.

```
poss(eintragen_Vollrecht(G,P,R,T),S) :- vollrecht(R),
       not eingetragenes_Recht(G,P,R,S).
poss(loeschen_Vollrecht(G,P,R,T),S) :- vollrecht(R),
       eingetragenes_Recht(G,P,R,S).

eingetragenes_Recht(G,P,R,do(A,S)) :- A=eintragen_Vollrecht(G,P,R,T);
       A=eintragen_Teilrecht_Wert(G,P,R,W,T);
       A=eintragen_Teilrecht_Nutzung(G,P,R,T);
       eingetragenes_Recht(G,P,R,S),
       not A=loeschen_Vollrecht(G,P,R,T),
       not A=loeschen_Teilrecht_Wert(G,P,R,W,T),
       not A=loeschen_Teilrecht_Nutzung(G,P,R,T).

proc( gb_transfer_Eigentum(G1,P1,P2 ),
       pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
       ?(eingetragenes_Recht(G1,P1,eigentum)):
       ?(dokument(G1,P1,P2,eigentum)):
       pi (t, (?(now(t))) : eintragen_Vollrecht(G1,P2,eigentum,t):
       loeschen_Vollrecht(G1,P1,eigentum,t): end_Grundbuchverfahren (G1,t))
).
```

Eigentumsrechte sind Vollrechte. Veränderungen werden abgebildet, indem Rechte eingetragen oder gelöscht werden. Ein Vollrecht kann eingetragen werden, wenn es noch nicht besteht. Nur bestehende Rechte können gelöscht werden. In Folge der Eintragung eines Rechts besteht dann eine Eintragung im Grundbuch (repräsentiert durch *eingetragenes_Recht*). Das Recht hört auf zu bestehen, wenn es gelöscht wird. Der Transfer eines Eigentumsrechts wird implementiert durch die GOLOG Prozedur *gb_transfer_Eigentum*. Dabei wird das Grundbuchverfahren ausgelöst. Es wird die Voreintragung des Eigentümers geprüft und das Vorliegen des Beweisdokuments für den Eigentumstransfer. Danach wird das Eigentumsrecht des Käufers eingetragen, das des alten Eigentümers gelöscht und abschließend das Grundbuchverfahren beendet.

6.5.7 Die Implementation der Inkonsistenzmarkierungen

Inkonsistenzmarkierungen zeigen an, an welchen Stellen materielles Recht und Grundbuch nicht in Übereinstimmung sind. Die Implementation soll demonstriert werden anhand der Markierung für inkonsistente Eigentumsverhältnisse. Diese Definition lautete:

$$Inkons\_Eigentümer(g,p,s) \Leftrightarrow$$
$$\neg(Eigentümer(g,p,s) \Leftrightarrow Eingetragenes\_Recht(g,p,Eigentum,s))$$

In dieser Form läßt sich diese Formel nicht in Prolog Syntax übersetzen, folglich müssen die Transformationsregeln angewendet werden, was zu der folgenden Definition führt:

$$Inkons\_Eigentümer(g,p,s) \Leftrightarrow$$
$$Eigentümer(g,p,s) \wedge \neg Eingetragenes\_Recht(g,p,Eigentum,s) \vee$$
$$\neg Eigentümer(g,p,s) \wedge Eingetragenes\_Recht(g,p,Eigentum,s)$$

Dies ist eine Formel, die sich in Prolog Syntax übersetzen läßt und die Übersetzung lautet:

```
inkonsE(G,P,S) :- eigentuemer(G,P,S), not
eingetragenes_Recht(G,P,eigentum,S);
       eingetragenes_Recht(G,P,eigentum,S), not eigentuemer(G,P,S).
```

Eine Nichtübereinstimmung von materiellem Recht und Grundbuch ergibt sich in zwei Fällen: Entweder ist eine Eintragung inkorrekt, oder sie ist im Grundbuch nicht vorhanden. Diese zwei Fälle bilden sich in der übersetzten Version der Inkonsistenzmarkierung ab. Eine Eintragung ist inkorrekt, also überflüssig, wenn ein Eigentumsrecht im Grundbuch eingetragen ist, aber keine materiellrechtliche Grundlage besteht.

```
inkons_eE(G,P,S) :- eingetragenes_Recht(G,P,eigentum,S),
       not eigentuemer(G,P,S).
```

Eine notwendige Eintragung fehlt im Grundbuch, wenn ein Recht materiellrechtlich besteht, aber keine Eintragung im Grundbuch vorhanden ist.

```
inkons_neE(G,P,S) :- eigentuemer(G,P,S),
       not eingetragenes_Recht(G,P,eigentum,S).
```

6.5.8 Implementation der Korrektur des unrichtigen Grundbuchs

Das Grundbuch kann dadurch berichtigt werden, daß eine inkorrekte Eintragung gelöscht oder eine fehlende Eintragung hinzugefügt wird. Nach Eingang einer Beschwerde gegen die Richtigkeit des Grundbuchs muß geprüft werden, ob für das betreffende Rechtsverhältnis eine Inkonsistenzmarkierung gesetzt ist oder nicht. Danach wird dem Grundbuch entweder eine Eintragung hinzugefügt oder eine Eintragung gelöscht. Die Beschwerde ist für das Grundbuchamt der Hinweis auf das Vorliegen einer Unrichtigkeit, die von Amts wegen beseitigt werden muß. Das Amt muß also auf das Vorliegen dieser Beschwerde und der damit zusammenhängenden Unrichtigkeit des Grundbuchs *reagieren*. Diese Vorgehensweise kann implementiert werden durch die Interruptsteuerung des RGOLOG Interpreters (Abschnitt 4.6.4). Die Idee war die Überwachung bestimmter Bedingungen und das Auslösen von Reaktionen, wenn diese Bedingungen eintreten. An dem Beispiel einer inkorrekten Eintragung eines Eigentumsrechts im Grundbuch soll die Implementation nun vorgestellt werden.

```
proc(interrupts,
       ((?(some(g,some(p,some(w, beschwerde(g,p,eigentum,w,loeschen)
              & inkons_eE(g,p)))))) :
              pi(g, pi(p, pi(w,
              (?(beschwerde(g,p,eigentum,w,loeschen) & inkons_eE(g,p))) :
              (pi(t, (?(now(t))) : loeschen_Vollrecht(g,p,eigentum,t) :
                     loeschen_Beschwerde(g,p,eigentum,w,loeschen,t)))))
       ) #
       (?(-(some(g,some(p,some(w, beschwerde(g,p,eigentum,w,eintragen)
              & inkons_neE(g,p)))))))
).
```

Wenn die Bedingung eintritt, daß eine Beschwerde über eine inkorrekte Eintragung im Grundbuch vorliegt und diese Eintragung als inkonsistent markiert ist, dann wird die betreffende Eintragung gelöscht.

6.5.9 Das Grundbuch als Auskunftssystem

Im Abschnitt 6.5.3 wurde bereits beschrieben, wie der aktuelle Inhalt der Datenbank bestimmt werden kann. Anfragen an die Datenbank müssen sich nicht nur auf die Werte von Fluents beziehen, sie können auch allgemeiner sein. Sie müssen allerdings in Prolog Syntax formulierbar sein, das heißt, daß es möglicherweise notwendig ist, die Anfrage erst mit Hilfe der Transformationsregeln in Prolog Syntax zu überführen. Grundsätzlich stellt sich die Forderung, daß nur Anfragen gestellt werden, für die Regression anwendbar ist, da nur für diese Klasse von Formeln die Korrektheit der Implementation garantiert ist.

6.6 Der Test der Implementation

Dieses Kapitel beinhaltet die Implementation des formalen Modells. Die Implementation liefert eine ausführbare Spezifikation. Der zentrale Punkt ist dabei, daß diese Implementation korrekt ist bezüglich des formalen Modells. Keine Aussagen wurden bislang über das Verhältnis dieses formalen Modells und des konzeptionellen Modells getroffen. Befinden sich Implementation und konzeptionelles Modell in Übereinstimmung? Bildet das formale Modell und damit die Implementation das konzeptionelle Modell korrekt ab? Dies sind die Fragen, die es nun zu analysieren gilt.
Es muß also geklärt werden, ob das Programm, also die ausführbare Spezifikation, die gewünschten, vom konzeptionellen Modell bestimmten Eigenschaften hat. Diese

Überprüfung kann erfolgen durch Testen (Sanella 1988). Anhand von geeigneten Beispielen wird überprüft, ob das Programm richtig funktioniert. Wie kann nun die Implementation des Grundbuchsystems getestet werden? Ausgangspunkt ist hierfür die Auswahl eines Testsszenarios, anhand dessen die Eigenschaften des Programms überprüft werden können. Hierbei ergeben sich drei Schwerpunkte:

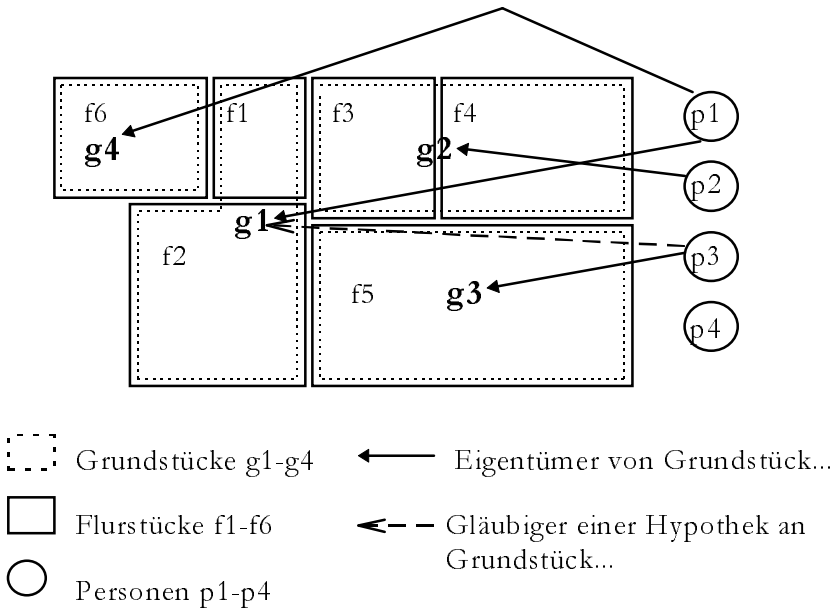
1. Bildet die Spezifikation die Arbeit des Grundbuchamts korrekt ab?
   Das heißt: Wird das Grundbuchrecht korrekt abgebildet? Werden die Abläufe, die im Grundbuchrecht festgelegt sind, korrekt abgebildet? In diesem Punkt wird überprüft, ob die Implementation die Abläufe, die im konzeptionellen Modell formuliert wurden, richtig umsetzt. In diesem Sinn wird getestet, ob der normale Ablauf der Arbeit des Grundbuchamts, wie sie das konzeptionelle Modell formuliert hat, richtig formal beschrieben wurde. Als normal werden die Abläufe insofern bezeichnet, daß keine Inkonsistenzen, also Unrichtigkeiten des Grundbuchs vorkommen. Zum Beispiel soll überprüft werden, ob nach einem im Grundbuch eingetragenen Eigentumstransfer die erwarteten Einträge im Grundbuch vorhanden sind.
2. Bildet die Spezifikation Inkonsistenzen zwischen materiellem Recht und Grundbuch korrekt ab?
   In diesem Punkt werden nach der Behandlung der normalen Funktionen des Grundbuchs die kritischen Fälle geprüft, in denen Unrichtigkeiten des Grundbuchs auftreten. Das sind die Fälle, die vom Grundbuch selbst nicht korrekt behandelt werden. Sie lassen sich in der Analyse der Beziehung der Modelle des Grundbuchs und des materiellen Rechts diskutieren.
3. Bildet die Spezifikation Funktionen korrekt ab, die Inkonsistenzen korrigieren?
   Hierbei werden die Funktionen überprüft, die zur Korrektur von Unrichtigkeiten im Grundbuch angegeben wurden.

Diese drei Punkte werden im Anschluß abgearbeitet, indem mit Hilfe des Testszenarios typische Fälle getestet werden.
Die Vorgehensweise ist dabei die folgende: Zuerst wird der zu modellierende Ablauf informell beschrieben. Danach erfolgt ein Programmlauf, der diesen Ablauf abbildet. Abschließend werden dann die Ergebnisses dieses Programmlaufs diskutiert.

6.6.1 Das Testszenario

Das Testszenario soll an der folgenden Grafik veranschaulicht werden:

Abbildung 6.1 Das Testszenario



Das Testszenario wird in die initiale Datenbank integriert und bestimmt dabei die Objekte, Rechtsverhältnisse und das Verhältnis von materiellem und formellem Recht. Die initiale Datenbank wird dementsprechend um den folgenden Inhalt erweitert:

```
%*******das materielle Recht*********
person(p1,s0).person(p2,s0).person(p3,s0).person(p4,s0).
grundstueck(g1,s0).grundstueck(g2,s0).grundstueck(g3,s0).
grundstueck(g4,s0).
flurstueck(f1,s0).flurstueck(f2,s0).flurstueck(f3,s0).
flurstueck(f4,s0).flurstueck(f5,s0).flurstueck(f6,s0).
grundstueck_Flurstuecke_m(g1,f1,s0).grundstueck_Flurstuecke_m(g1,f2,s0).
grundstueck_Flurstuecke_m(g2,f3,s0).grundstueck_Flurstuecke_m(g2,f4,s0).
grundstueck_Flurstuecke_m(g3,f5,s0).grundstueck_Flurstuecke_m(g4,f6,s0).
eigentuemer(g1,p1,s0).eigentuemer(g2,p2,s0).eigentuemer(g3,p3,s0).
eigentuemer(g4,p1,s0).
glaeubiger_Hypothek(g1,p3,1000,s0).
%********Der Inhalt des Grundbuches***********
eingetragenes_Recht(g1,p1,eigentum,s0).
eingetragenes_Recht(g2,p2,eigentum,s0).
eingetragenes_Recht(g3,p3,eigentum,s0).
eingetragenes_Recht(g4,p1,eigentum,s0).
eingetragenes_Recht(g1,p3,hypothek,s0).
wert_Recht(g1,p3,hypothek,1000,s0).
grundstueck_Flurstuecke(g1,f1,s0).
grundstueck_Flurstuecke(g1,f2,s0).
grundstueck_Flurstuecke(g2,f3,s0).
grundstueck_Flurstuecke(g2,f4,s0).
grundstueck_Flurstuecke(g3,f5,s0).
grundstueck_Flurstuecke(g4,f6,s0).
```

Materielles Recht und der Inhalt des Grundbuchs stehen in Übereinstimmung. Dieses Szenario soll nun Grundlage für den Test der Spezifikation sein.

6.6.2 Der normale Ablauf

Normale Abläufe in dem oben beschriebenen Sinn sind alle Abläufe zur Erreichung von Eintragungen in das Grundbuch ohne daß dabei Inkonsistenzen auftreten. Dazu gehören zum Beispiel Eigentumstransfers, die Eintragung oder Löschung einer Hypothek oder die Eintragung einer Grundstücksteilung im Grundbuch. Als Beispiel für einen Test soll das folgende Beispiel (innerhalb des oben beschriebenen Testszenarios) dienen.
Die Person p1 will das Flurstück f2 verkaufen, das Bestandteil seines Grundstücks g1 ist. Dazu teilt sie das Flurstück ab und läßt diese Teilung in das Grundbuch eintragen. Das abgeteilte Grundstück erhält die Bezeichnung g5. Anschließend erfolgt der Verkauf dieses Grundstücks an die Person p2. p2 läßt den Eigentumstransfer im Grundbuch eintragen und legt hierfür den Kaufvertrag als Beweis beim Grundbuchamt vor.

```
ECLiPSe Constraint Logic Programming System [sepia mps]
Version 3.7.1, Copyright ECRC GmbH and ICL/IC-Parc, Fri Mar 20 15:51 1998
[eclipse 1]: gamt.
Enter an exogenous action, or nil. teile_Grundstueck(g1, g5, p1, f2, 10).
Enter an exogenous action, or nil. register_Antrag(teilung_e, g1, g5, f2, p1, nix,
nix, 20).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. verkaufe_Grundstueck(g5,p1,p2,30).
Enter an exogenous action, or nil. register_Antrag(transfer_e, g5, nix, nix, p1, p2,
nix, 40).
Enter an exogenous action, or nil. register_Dokument(g5, p1, p2, eigentum, 50).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. stop(60).
Enter an exogenous action, or nil. nil.
```

```
[wait(1), teile_Grundstueck(g1, g5, p1, f2, 10), register_Antrag(teilung_e, g1, g5,
f2, p1, nix, nix, 20), start_Grundbuchverfahren(g1, 20),
eintragen_teilen_Grundstueck(g1, g5, f2, 20), start_Grundbuchverfahren(g5, 20),
eintragen_Vollrecht(g5, p1, eigentum, 20), end_Grundbuchverfahren(g1, 20),
end_Grundbuchverfahren(g5, 20), loesche_Antrag(teilung_e, g1, g5, f2, p1, nix, nix,
20), verkaufe_Grundstueck(g5, p1, p2, 30), register_Antrag(transfer_e, g5, nix, nix,
p1, p2, nix, 40), register_Dokument(g5, p1, p2, eigentum, 50),
start_Grundbuchverfahren(g5, 50), eintragen_Vollrecht(g5, p2, eigentum, 50),
loeschen_Vollrecht(g5, p1, eigentum, 50), end_Grundbuchverfahren(g5, 50),
loesche_Antrag(transfer_e, g5, nix, nix, p1, p2, nix, 50), wait(50), stop(60)]

materielles Recht:
Grundstuecke: [g5, g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g5, f2), (g1, f1), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g5, p2), (g1, p1), (g2, p2), (g3, p3), (g4, p1)]
Glaeubiger von Hypotheken: [(g1, p3, 1000)]
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g5, p2, eigentum), (g1, p1, eigentum), (g2, p2, eigentum),
(g3, p3, eigentum), (g4, p1, eigentum), (g1, p3, hypothek)]
Werte von Rechten: [(g1, p3, hypothek, 1000)]
Grundstücke/Flurstuecke: [(g5, f2), (g1, f1), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: [(g5, p1, p2, eigentum)]

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: []
inkonsistente Hypotheken: []
inkonsistente Nieszbraeuche: []
inkonsistente Grundstuecke/Flurstuecke: []


yes.
[eclipse 2]:
```

Aus dem Output des Programmlaufs geht hervor, daß nach diesem Ablauf das Grundstück g5 neu vorhanden ist, das aus dem Flurstück f2 besteht. Als Eigentümer des Grundstücks g5 ist p2 eingetragen. Es bestehen keine Inkonsistenzen zwischen materiellem Recht und Grundbuch. Dies ist auch genau das, was erreicht werden sollte und entspricht den im konzeptionellen Modell beschriebenen Abläufen zur Grundstücksteilung und zum Eigentumstransfer.

6.6.3 Die Markierung von Inkonsistenzen

In diesem Abschnitt sollen drei Fälle getestet werden, in denen das Grundbuch unrichtig wird. Wichtig ist hierbei die zuverlässige Kennzeichnung der Inkonsistenzen. Die vorgestellten Fälle entsprechen denen, die in Abschnitt 5.4.3 eingeführt wurden.

- Die Unrichtigkeit des Grundbuchs infolge Rechtsübergang außerhalb des Grundbuchs:

Zu Gunsten der Person p4 wird ein Nießbrauch am Grundstück g2 bestellt und im Grundbuch eingetragen. Anschließend stirbt die Person p4.

```
[eclipse 3]: gamt.
Enter an exogenous action, or nil. bestelle_Nieszbrauch(g2,p2,p4,10).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil.
register_Antrag(nieszbrauch_e,g2,nix,nix,p4,nix,nix, 20).
Enter an exogenous action, or nil. register_Dokument(g2,p2,p4,nieszbrauch,30).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. stirbt_Person(p4,40).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. stop(50).
Enter an exogenous action, or nil. nil.

[wait(1), bestelle_Nieszbrauch(g2, p2, p4, 10), wait(10),
register_Antrag(nieszbrauch_e, g2, nix, nix, p4, nix, nix, 20), register_Dokument(g2,
p2, p4, nieszbrauch, 30), start_Grundbuchverfahren(g2, 30),
eintragen_Teilrecht_Nutzung(g2, p4, nieszbrauch, 30), end_Grundbuchverfahren(g2, 30),
```

```
loesche_Antrag(nieszbrauch_e, g2, nix, nix, p4, nix, nix, 30), stirbt_Person(p4, 40),
wait(40), stop(50)]

materielles Recht:
Grundstuecke: [g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g1, p1), (g2, p2), (g3, p3), (g4, p1)]
Glaeubiger von Hypotheken: [(g1, p3, 1000)]
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g2, p4, nieszbrauch), (g1, p1, eigentum), (g2, p2, eigentum),
(g3, p3, eigentum), (g4, p1, eigentum), (g1, p3, hypothek)]
Werte von Rechten: [(g1, p3, hypothek, 1000)]
Grundstücke/Flurstuecke: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: [(g2, p2, p4, nieszbrauch)]

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: []
inkonsistente Hypotheken: []
inkonsistente Nieszbraeuche: [(g2, p4)]
inkonsistente Grundstuecke/Flurstuecke: []


yes.
[eclipse 4]:
```

Durch den Tod der Person p4 wird das Grundbuch unrichtig, da der Nießbrauch ein Recht ist, das auf die Lebenszeit der Person beschränkt ist, zu deren Gunsten er besteht. Materiellrechtlich erlischt dieses Recht also mit dem Tod der Person. Das Grundbuch wird dadurch unrichtig, daß noch immer der Nießbrauch im Grundbuch eingetragen ist. Durch das Programm wird diese Inkonsistenz gekennzeichnet, folglich hat es sich korrekt verhalten.

- Verletzung der Verfahrensregeln durch die Beteiligten:

Problematisch für das Grundbuchrecht sind Situationen, in denen die Gefahr gutgläubigen Erwerbs besteht. Folgende Situation ist denkbar: Person p1 verkauft das Grundstück g4 an die Person p4. p4 versäumt es, sein Recht im Grundbuch eintragen zu lassen. p1 verkauft das Grundstück ein zweites Mal an p3. Dieser Eigentumsübergang kommt materiellrechtlich nicht zustande, da ja p1 nicht mehr Eigentümer ist. p3 kann nun allerdings mit dem Kaufvertrag zum Grundbuchamt gehen und das Eigentumsrecht eintragen lassen, da das Grundbuchamt nur das Vorhandensein des Beweisdokuments prüft, nicht aber die materiellen Voraussetzungen. Das Grundbuch wird mit dieser Eintragung unrichtig, da die materielle Rechtslage nicht in Übereinstimmung mit dem Grundbuch ist.

```
[eclipse 5]: gamt.
Enter an exogenous action, or nil. verkaufe_Grundstueck(g4,p1,p2,10).
Enter an exogenous action, or nil.
register_Antrag(transfer_e,g4,nix,nix,p1,p3,nix,20).
Enter an exogenous action, or nil. register_Dokument(g4,p1,p3,eigentum,30).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. stop(40).
Enter an exogenous action, or nil. nil.

[wait(1), verkaufe_Grundstueck(g4, p1, p2, 10), register_Antrag(transfer_e, g4, nix,
nix, p1, p3, nix, 20), register_Dokument(g4, p1, p3, eigentum, 30),
start_Grundbuchverfahren(g4, 30), eintragen_Vollrecht(g4, p3, eigentum, 30),
loeschen_Vollrecht(g4, p1, eigentum, 30), end_Grundbuchverfahren(g4, 30),
loesche_Antrag(transfer_e, g4, nix, nix, p1, p3, nix, 30), wait(30), wait(30),
stop(40)]

materielles Recht:
Grundstuecke: [g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g4, p2), (g1, p1), (g2, p2), (g3, p3)]
```

```
Glaeubiger von Hypotheken: [(g1, p3, 1000)]
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g4, p3, eigentum), (g1, p1, eigentum), (g2, p2, eigentum),
(g3, p3, eigentum), (g1, p3, hypothek)]
Werte von Rechten: [(g1, p3, hypothek, 1000)]
Grundstücke/Flurstuecke: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: [(g4, p1, p3, eigentum)]

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: [(g4, p2), (g4, p3)]
inkonsistente Hypotheken: []
inkonsistente Nieszbraeuche: []
inkonsistente Grundstuecke/Flurstuecke: []


yes.
[eclipse 6]:
```

Nach dem beschriebenen Ablauf ist das Grundbuch an zwei Stellen unrichtig. Einerseits ist das Eigentumsrecht an Grundstück g4 zu Gunsten von p3 eingetragen, wofür es keine materiellrechtliche Grundlage gibt. Andererseits fehlt die Eintragung von p2 als Eigentümer von g4, die die materielle Rechtslage erfordert. Im konventionellen Grundbuchsystem würde die Eintragung zu Gunsten von p3 als richtig gelten, p2 müßte entschädigt werden. In diesem Modell kann der gutgläubige Erwerb von p3 vermieden werden, da die Nichtübereinstimmung von Grundbuch und materieller Rechtslage bekannt, also markiert ist. Folglich ist der gute Glaube der Person p3 zerstört. Also könnte die unrichtige Eintragung von p3 als Eigentümer unterbleiben und damit auch der finanzielle Schaden durch die Entschädigung des wirklichen Eigentümers p2.

- Verwechslung einer Flurstücksnummer bei der Grundstücksteilung

Die Person g1 will das Flurstück f1 verkaufen, das Teil seines Grundstücks g1 ist. f1 wird von g1 abgeteilt und danach als Grundstück g5 geführt. Bei der Eintragung dieser Grundstücksteilung im Grundbuch werden die Bezeichnungen der Flurstücke verwechselt. Versehentlich wird Flurstück f2 abgeteilt und als Grundstück g5 im Grundbuch eingetragen. Person p1 verkauft nun Grundstück g5 an Person p2. Dieser läßt sein Recht im Grundbuch eintragen. Das Grundbuch ist nun unrichtig, da das Grundstück g5 aus dem falschen Flurstück gebildet wurde und folglich p2 als Eigentümer eines Stücks Land im Grundbuch eingetragen ist, das er nicht erwerben wollte. Die materiellrechtlichen Grundlagen des Verkaufs sind also nicht gegeben, hat man sich doch über den Gegenstand des Rechtsgeschäfts geirrt.

```
[eclipse 5]: gamt.
Enter an exogenous action, or nil. teile_Grundstueck(g1, g5, p1, f1, 10).
Enter an exogenous action, or nil. register_Antrag(teilung_e, g1, g5, f2, p1, nix,
nix, 20).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. verkaufe_Grundstueck(g5,p1,p2,30).
Enter an exogenous action, or nil.
register_Antrag(transfer_e,g5,nix,nix,p1,p2,nix,40).
Enter an exogenous action, or nil. register_Dokument(g5,p1,p2,eigentum,50).
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. nil.
Enter an exogenous action, or nil. stop(60).
Enter an exogenous action, or nil. nil.

[wait(1), teile_Grundstueck(g1, g5, p1, f1, 10), register_Antrag(teilung_e, g1, g5,
f2, p1, nix, nix, 20), start_Grundbuchverfahren(g1, 20),
eintragen_teilen_Grundstueck(g1, g5, f2, 20), start_Grundbuchverfahren(g5, 20),
eintragen_Vollrecht(g5, p1, eigentum, 20), end_Grundbuchverfahren(g1, 20),
end_Grundbuchverfahren(g5, 20), loesche_Antrag(teilung_e, g1, g5, f2, p1, nix, nix,
```

```
20), verkaufe_Grundstueck(g5, p1, p2, 30), register_Antrag(transfer_e, g5, nix, nix,
p1, p2, nix, 40), register_Dokument(g5, p1, p2, eigentum, 50),
start_Grundbuchverfahren(g5, 50), eintragen_Vollrecht(g5, p2, eigentum, 50),
loeschen_Vollrecht(g5, p1, eigentum, 50), end_Grundbuchverfahren(g5, 50),
loesche_Antrag(transfer_e, g5, nix, nix, p1, p2, nix, 50), stop(60)]

materielles Recht:
Grundstuecke: [g5, g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g5, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g5, p2), (g1, p1), (g2, p2), (g3, p3), (g4, p1)]
Glaeubiger von Hypotheken: [(g1, p3, 1000)]
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g5, p2, eigentum), (g1, p1, eigentum), (g2, p2, eigentum),
(g3, p3, eigentum), (g4, p1, eigentum), (g1, p3, hypothek)]
Werte von Rechten: [(g1, p3, hypothek, 1000)]
Grundstücke/Flurstuecke: [(g5, f2), (g1, f1), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: [(g5, p1, p2, eigentum)]

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: []
inkonsistente Hypotheken: []
inkonsistente Nieszbraeuche: []
inkonsistente Grundstuecke/Flurstuecke: [(g5, f1), (g1, f2), (g5, f2), (g1, f1)]


yes.
[eclipse 6]:
```

Durch den Programmlauf werden vier inkonsistente Stellen markiert. Einerseits die Zuordnung von f2 zu g5 und f1 zu g1. Dies ist die Lage, die aus dem Grundbuch hervorgeht. Da die Flurstücksnummern verwechselt wurden, sind diese Eintragungen inkorrekt, also überflüssig. Andererseits sind die Zuordnungen von f1 zu g5 und f2 zu g1 als inkonsistent markiert, da diese Zuordnungen durch die materielle Rechtslage gegeben ist, aber nicht im Grundbuch eingetragen wurde.

6.6.4 Die Korrektur des unrichtigen Grundbuchs

Das Grundbuch kann dadurch korrigiert werden, daß inkorrekte Eintragungen gelöscht oder das Grundbuch um fehlende Eintragungen ergänzt wird. Um die Funktion der korrigierenden Transaktionen einfach zu testen, wird das Testszenario verändert.
Dazu erfolgt die Änderung der initialen Datenbank, indem die folgenden Klauseln entfernt werden:

```
glaeubiger_Hypothek(g1,p3,1000,s0).
eingetragenes_Recht(g4,p1,eigentum,s0).
```

Dadurch wird das Grundbuch unrichtig. Einerseits ist in ihm nach wie vor ein Recht über eine Hypothek eingetragen, obwohl die materiellrechtliche Grundlage entfernt wurde. Das bedeutet, daß dieses Recht als überflüssig markiert und aus dem Grundbuch gelöscht werden muß. Andererseits wurde die Eintragung über ein Eigentumsrecht aus dem Grundbuch entfernt, obwohl dieses Recht besteht. Dieses Recht muß als fehlend im Grundbuch markiert sein. Dazu wird der Inhalt der initialen Datenbank ausgegeben:

```
[eclipse 1]: printFluentList(s0).

materielles Recht:
Grundstuecke: [g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g1, p1), (g2, p2), (g3, p3), (g4, p1)]
Glaeubiger von Hypotheken: []
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g1, p1, eigentum), (g2, p2, eigentum), (g3, p3, eigentum),
(g1, p3, hypothek)]
```

```
Werte von Rechten: [(g1, p3, hypothek, 1000)]
Grundstücke/Flurstuecke: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: []

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: [(g4, p1)]
   Einträge unrichtg: []
   Einträge nicht vorhanden: [(g4, p1)]
inkonsistente Hypotheken: [(g1, p3, 1000)]
   Einträge unrichtg: [(g1, p3, 1000)]
   Einträge nicht vorhanden: []
inkonsistente Nieszbraeuche: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []
inkonsistente Grundstuecke/Flurstuecke: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []


yes.
[eclipse 2]:
```

Die Darstellung der Inkonsistenzen wurde verfeinert, indem jetzt nicht nur die allgemeine Inkonsistenzmarkierung ausgegeben wird, sondern auch die Fluents, die fehlende und überflüssige Eintragungen markieren (siehe Abschnitt 6.5.8). Es sind, wie es erwartet wurde, die Eintragungen als fehlend bzw. als inkorrekt markiert. Gegen beide Eintragungen wird Beschwerde eingelegt und das Grundbuch bezüglich der fehlenden Eintragung ergänzt sowie die überflüssige Eintragung gelöscht.

```
[eclipse 1]: gamt.
Enter an exogenous action, or nil. register_Beschwerde(g4, p1, eigentum, nix,
eintragen, 10).
Enter an exogenous action, or nil.
register_Beschwerde(g1,p3,hypothek,1000,loeschen,20).
Enter an exogenous action, or nil. stop(30).

[wait(1), register_Beschwerde(g4, p1, eigentum, nix, eintragen, 10), wait(10),
eintragen_Vollrecht(g4, p1, eigentum, 10), loeschen_Beschwerde(g4, p1, eigentum, nix,
eintragen, 10), register_Beschwerde(g1, p3, hypothek, 1000, loeschen, 20), wait(20),
loeschen_Teilrecht_Wert(g1, p3, hypothek, 1000, 20), loeschen_Beschwerde(g1, p3,
hypothek, 1000, loeschen, 20), stop(30)]

materielles Recht:
Grundstuecke: [g1, g2, g3, g4]
Personen: [p1, p2, p3, p4]
Grundstücke/Flurstuecke/m: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4,
f6)]
Eigentuemer: [(g1, p1), (g2, p2), (g3, p3), (g4, p1)]
Glaeubiger von Hypotheken: []
Nieszbraucher: []

Grundbuch:
eingetragene Rechte: [(g4, p1, eigentum), (g1, p1, eigentum), (g2, p2, eigentum),
(g3, p3, eigentum)]
Werte von Rechten: []
Grundstücke/Flurstuecke: [(g1, f1), (g1, f2), (g2, f3), (g2, f4), (g3, f5), (g4, f6)]
Dokumente: []

Inkonsistenzen:
inkonsistente Eigentumsverhältnisse: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []
inkonsistente Hypotheken: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []
inkonsistente Nieszbraeuche: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []
inkonsistente Grundstuecke/Flurstuecke: []
   Einträge unrichtg: []
   Einträge nicht vorhanden: []


yes.
[eclipse 2]:
```

Im Ergebnis des Programmlaufs wurden die Inkonsistenzen beseitigt.

6.7 Zusammenfassung

Dieses Kapitel beinhaltete die Erstellung der Implementation eines Grundbuchsystems. Die formalen Grundlagen der Implementation begründeten eine sehr geradlinige Übersetzung der Spezifikation in ein Prolog Programm. Für eine große Klasse von Formeln reichte es aus, die logischen Operatoren durch ihre entsprechenden Prolog Operatoren zu ersetzen. Die in dieser Arbeit erstellte Spezifikation bestand bis auf eine Ausnahme aus Formeln, die sich so direkt in Prolog Syntax übersetzen ließen. Lediglich die Definitionen der Inkonsistenzmarkierungen erforderte die Anwendung einiger Transformationsregeln, um eine in Prolog übersetzbare syntaktische Form zu erreichen. Auf diesem einfachen Zusammenhang zwischen der Spezifikation und dem zugehörenden Prolog Programm beruht die Sichtweise, eine solche Spezifikation von Datenbank Transaktionen als ausführbare Spezifikation zu betrachten. Das Prolog Programm bildete die Grundlage für den Test der Spezifikation, also die Beantwortung der Frage, ob die Spezifikation den zu modellierenden Teil der Welt, das konzeptionelle Modell, korrekt abbildet. Die Darstellung dieser Tests mußte sich in dieser Arbeit auf einige wenige interessante Fälle in einem bestimmten Szenario beschränken.

In der Implementation wurden der sequentielle temporale GOLOG Interpreter und der RGOLOG Interpreter miteinander verknüpft. Dies ermöglichte einerseits die notwendige Abbildung von zeitlichen Abläufen des Grundbuchs und andererseits die Beschreibung von Ereignissen, die durch Abläufe in der materiellen Welt verursacht wurden, ohne diese Abläufe selbst spezifizieren zu müssen.

Die Implementation der materiellen Rechtsverhältnisse und des Grundbuchs ermöglichte es, in Abhängigkeit von diesen Ereignissen in der materiellen Welt, Nichtübereinstimmungen zwischen diesen Modellen automatisch zu markieren. Diese Markierungen bildeten die Grundlage für die Implementation korrigierender Transaktionen.

7 Zusammenfassung

7.1 Was wurde getan?

Die große Bedeutung von Grund und Boden verursacht ein erhebliches Interesse an einer zuverlässigen Verwaltung und Offenlegung der Eigentumsverhältnisse an Grundstücken. Historisch entwickelten sich zu diesem Zweck Grundbücher. Diese werden mit hohem Aufwand gepflegt, was die Notwendigkeit einer effektiven Verwaltung zur Folge hat. Daher gibt es verstärkt Bestrebungen, Grundbücher in Datenbank- und Geographische Informationssysteme zu integrieren. Inhalt dieser Arbeit war es deshalb, einen Beitrag zu leisten bei der Analyse der Probleme, die bei der Repräsentation und Implementation eines solchen Systems auftreten. Ausgehend von dieser Analyse wurde ein Ausschnitt aus einem Grundbuchsystem modelliert und anschließend eine prototypische Implementation erstellt. Die Erstellung des Modells und seine Implementation erfolgte auf Basis des Situationskalküls, der in der Künstlichen Intelligenz ein sehr verbreiteter Ansatz ist. Die Verwendung dieses Ansatzes unterstreicht den interdisziplinären Charakter vieler Probleme und zeigt Verbindungen zwischen dem Fachgebiet der Künstlichen Intelligenz, der Datenbanktheorie und der Software Entwicklung.
Der interdisziplinäre Charakter der Modellierung eines Grundbuchsystems findet seinen Ausdruck in den Ergebnissen, die nun im einzelnen formuliert werden sollen.

7.2 Ergebnisse

7.2.1 Räumliche und zeitliche Veränderungen

Aus der Analyse des Grundbuches ergab sich die entscheidende Bedeutung des Verständnisses der räumlichen und zeitlichen Veränderungen für die Beschreibung eines solchen Systems. So spielt die Modellierung von zeitlichen und räumlichen Abläufen, die diese Veränderungen verursachen, eine zentrale Rolle. Diese Arbeit unterstützt die Ansicht, daß ein System nicht als statisch und Veränderungen nicht als Übergang zwischen statischen Zuständen aufgefaßt werden können, sondern diese Veränderungen und die sie verursachenden Abläufe selbst das System beschreiben.

7.2.2 Grundbuch und materielle Rechtsverhältnisse

Bei der Beschreibung des Grundbuchsystems brachte die Unterscheidung von materieller Welt und Grundbuch entscheidende Vorteile. Grundbücher wurden dafür geschaffen, eben diese materiellen Rechtsverhältnisse abzubilden. Es bildet in diesem Sinne eine Repräsentation dieser materiellen Rechtsverhältnisse. Ein zentrales Problem einer jeden Repräsentation von Phänomenen in der Welt ist die Tatsache, daß diese Repräsentation unvollständig bleiben muß. So ergab auch die Analyse des Grundbuches, daß aus der Nichtübereinstimmung von Grundbuch und materiellem Recht große Schwierigkeiten entstehen. Diese Schwierigkeiten haben einen beträchtlichen praktischen Stellenwert, verursachen sie doch große finanzielle Schäden in Grundstücksverkehr. Um die Nichtübereinstimmung von Grundbuch und materiellen Rechtsverhältnissen, die Grundbuchunrichtigkeit, diskutieren zu können, wurden sowohl Grundbuch als auch das materielle Recht modelliert. Das Ergebnis dieses Ansatzes war eine sehr unkomplizierte Kennzeichnung von unrichtigen Stellen des Grundbuchs.
Wichtig ist an dieser Stelle die Einsicht, daß das Modell der materiellen Rechtsverhältnisse wieder unzulänglich und unvollständig ist. Trotzdem ermöglicht die Modellierung der materiellen Rechtsverhältnisse ein besseres Verständnis der auftretenden Probleme dadurch, daß sie diese Probleme einer formalen Beschreibung zuführt. Die praktische Bedeutung liegt darin, daß man die Veränderungen im Modell der materiellen Rechtsverhältnisse als

Beobachtungen von Veränderungen in der materiellen Welt sehen kann. Diese Beobachtungen setzt man nun in Beziehung zu der Repräsentation dieser Welt. Dies ermöglicht das Erkennen von Problemen und Unzulänglichkeiten. Weiterhin zeigt es Wege auf, diese Unzulänglichkeiten zu korrigieren. In dieser Arbeit erfolgte dies durch die Definition von Operationen, die die Unrichtigkeit des Grundbuchs korrigieren.

7.2.3 Wissensrepräsentation und Spezifikation

Bei der formalen Umsetzung des Modells wurden Zusammenhänge zwischen dem Gebiet der Wissensrepräsentation und dem der formalen Spezifikation offenbar. Wissensrepräsentation beschäftigt sich mit der Beschreibung von Wissen über die materielle Welt. Formale Spezifikation beschreibt die Eigenschaften von Programmen. In Bezug auf das Grundbuchsystem bedeutet dies, daß aus der Sicht der Wissensrepräsentation Wissen über die materiellen Rechtsverhältnisse modelliert wird. Aus der Sicht der formalen Spezifikation kann das Modell als Spezifikation der Implementation eines Grundbuchsystems aufgefaßt werden. Nun ist es aber so, daß das Modell in beiden Fällen das gleiche ist, lediglich die Fragestellung unterscheidet sich. Daraus ergibt sich die Möglichkeit, aus dem Modell etwas über die materiellen Rechtsverhältnisse zu lernen und andererseits Grundlagen für eine Implementation zu legen. Dies unterstreicht einerseits den interdisziplinären Charakter des Problems und spannt andererseits den Bogen zwischen theoretischen Problemen und ihrer praktischen Umsetzung.

7.2.4 Das Modell des Grundbuches als Datenbankspezifikation

Die praktische Bedeutung dieser Arbeit besteht darin, daß das erstellte Modell des Grundbuchsystems als Datenbankspezifikation aufgefaßt werden kann. Darin liegt der direkte Zusammenhang zur Implementation des Grundbuches innerhalb eines Geographischen Informationssystems. Das Modell ist die Spezifikation der Implementation des Grundbuches in diesem Informationssystem. Die prototypische Implementation, die auf Basis dieses Modells erstellt werden kann, bildet das Werkzeug, um die Korrektheit dieser Datenbankimplementation zu überprüfen.

7.3 Was bleibt zu tun?

Mit der Einschränkung auf das Grundbuch ohne die Beschreibung des Katasters wurde der Modellierung von zeitlichen Veränderungen und zeitlichen Abläufen Vorrang vor räumlichen Abläufen gegeben. Der Untersuchung dieser räumlichen Zusammenhänge kommt trotzdem eine große Bedeutung zu. Aus der Analyse dieser Zusammenhänge sind sicher interessante Ergebnisse bezüglich der Natur räumlicher Abläufe zu erwarten. Diese Aufgabe bleibt zukünftiger Arbeit überlassen.
Das erstellte Modell bildet nur einen sehr beschränkten Ausschnitt aus einem Grundbuch ab. Interessant ist die Erweiterung dieses Modells als Basis für eine vollständige Implementation. In dieser Arbeit wird die Sicht vertreten, daß die prototypische Implementation als ausführbare Spezifikation angesehen werden kann. Nun kann dieser Prototyp auch selbst weiterentwickelt werden zu einer praktisch anwendbaren Implementation. Anfänge in diese Richtung wurden bereits gemacht (Bertossi, Arenas et al. 1996). In direktem Zusammenhang zu diesem Gedanken steht auch die folgende Sichtweise: Die Implementation des Modells in Prolog eröffnet eine direkte Verbindung zu der Betrachtung von Datenbanken aus der Sicht des logischen Programmierens (Gallaire, Minker et al. 1984). Die Untersuchung dieses Zusammenhanges bleibt der Zukunft überlassen.

7.4 Schlußfolgerungen

Die Darstellung der Ergebnisse zielte in zwei Richtungen. Einerseits wurde ein Beitrag geleistet zur Ermöglichung einer konkreten Implementation eines Grundbuchsystems.

Andererseits ist diese Arbeit auch Grundlage für die Betrachtung allgemeinerer Probleme. So ist das Verständnis von räumlichen und zeitlichen Veränderungen und die Modellierung von Wissen darüber von zentraler praktischer und theoretischer Bedeutung. Grundbuchsysteme sind für die Betrachtung auch solcher allgemeiner Fragen geeignet, weil die dabei auftretenden Probleme sehr vielfältig sind, und es sich trotzdem um einen relativ abgeschlossenen und gut abzugrenzenden Bereich handelt. Dies eröffnet die Möglichkeit, aus der Betrachtung des speziellen Problems etwas über die allgemeinen Zusammenhänge zu lernen. Der direkte Zusammenhang zu praktischen Problemen in Gestalt der Implementation eines Grundbuchsystems ermöglicht es, den Bogen zu spannen zwischen den theoretischen Problemen und der praktischen Umsetzung, und aus der Betrachtung des einen für das andere Erkenntnisse zu gewinnen.

8 Literatur

K. Al-Taha (1992). Temporal Reasoning in Cadastral Systems. Department of Surveying Engineering. Orono, University of Maine.

M. Arenas und L. Bertossi (1998). Specifying the Evolution of Relational Views- Extended Version, letzter Zugriff am: 22.09.98, URL=http://dcc.ing.puc.cl/~bertossi/

M. Bengel und F. Simmerding (1995). Grundbuch, Grundstück, Grenze: Handbuch zur Grundbuchordung unter Berücksichtigung katasterrechtlicher Fragen. Neuwied, Kriftel, Berlin, Hermann Luchterhand Verlag GmbH.

L. Bertossi, M. Arenas und C. Ferretti (1996). SCDBR: An automated Reasoner for Database Updates. XVI International Conference of the Chilean Computer Science Society, Valdivia.

BGB (1987). Das Bürgerliche Gesetzbuch. München, Verlag C.H.Beck.

A. Borgia, J. Mylopoulos und R. Reiter (1995). "On the Frame Problem in Procedure Specifications." IEEE Transactions on Software Engineering **21**(10): 785-798.

I. Bratko (1990). PROLOG- Programming for Artificial Intelligence, Addison-Wesley.

K. I. Clark (1978). Negation as failure. Logic and Data Bases. H. Gallaire und J. Minker. New York, Plenum Press: 292-322.

E. M. Clarke und J. M. Wing (1996). Formal Methods: State of the Art and Future Directions, CMU Computer Science.

COROBO (1998). GOLOG Interpreter, letzter Zugriff am: 7.5.98, URL=http://www.cs.toronto.edu/~corobo/gologinterpreter

J. Demharter (1997). Grundbuchordnung. München, C.H. Beck'sche Verlagsbuchhandlung.

D. Eickmann (1986). Grundbuchverfahrensrecht. Bielefeld, Verlag Ernst und Werner Gieseking.

A. U. Frank (1996). An Object-Oriented, Formal Approach to the Design of Cadastral Systems. SDH, Delft, The Netherlands, Taylor & Francis.

H. Gallaire, J. Minker und J.-M. Nicolas (1984). "Logic and Databases: A Deductive Approach." Computing Surveys **16**(2).

GBO (1987-1995). Grundbuchordnung. Grundbuch Grundstück Grenze, Handbuch zur Grundbuchordnung unter Berücksichtigung katasterrechtlicher Fragen. Manfred Bengel und F. Simmerding. Neuwied, Kriftel, Berlin, Hermann Luchterhand Verlag GmbH: 3-49.

IC-Parc (1996-1998). ECLiPSe- The ECRC Constraint Logic Parallel System, letzter Zugriff am: 09.07.98, URL=http://www.icparc.ic.ac.uk/eclipse/

IC-Parc (1998). ECLiPSe- Extensions User Manual, Release 3.7, letzter Zugriff am: 09.07.98, URL=http://www.icparc.ic.ac.uk/eclipse

R. Laurini und D. Thompson (1992). Fundamentals of Spatial Information Systems, Academic Press.

H. J. Levesque, R. Reiter, Y.Lesperance, F. Lin und R. Scherl (1996). "GOLOG: A logic Programming Language for Dynamic Domains." Journal of Logic Programming, Special Issue on Reasoning about Action and Change.

H. Löffler (1993). Grundbuch und Grundstücksrecht. Frankfurt am Main, Fritz Knapp Verlag.

J. McCarthy und P. J. Hayes (1969). Some Philosophical Problems from the Standpoint of Artificial Intelligence. Machine Intelligence 4. B. Melzer und D. Michie. Edinburgh, Edinburg University Press: 463-502.

S. A. McIlraith (1997). A Closed-Form Solution to the Ramification Problem (Sometimes). Proceedings of the IJCAI'97 Workshop on Nonmonotonic Reasoning Action and Change.

J. A. Pinto und R. Reiter (1995). "Reasoning about Time in the Situation Calculus." Annals of Mathematics and Artificial Intelligence **14**(2-4): 251-268.

R. Reiter (1984). Towards a Logical Reconstruction of Relational Database Theory. On Conceptual Modelling, Perspectives from Artificial Intelligence, Databases, and Programming Languages. M. L. Brodie, J. Mylopoulos und J. W. Schmidt, Springer-Verlag.

R. Reiter (1991). The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy. V. Lifschitz. San Diego, Academic Press: 359-380.

R. Reiter (1992a). Formalizing Database Evolution in the Situation Calculus. Int. Conf. on Fifth Generation Computer Systems, Tokyo, Japan.

R. Reiter (1992b). In formalizing database updates: preliminary report. 3rd Int. Conf. on Extending Database Technology, Vienna, Austria.

R. Reiter (1992c). The projection Problem in the Situation Calculus: a Soundness and Completeness Result, with an Application to Database Updates. First Int. Conference on AI Planning Systems, College Park, Maryland.

R. Reiter (1995). "On specifying database updates." Journal of Logic Programming **25**(1): 53-91.

R. Reiter (1998a). Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems, DRAFT.

R. Reiter (1998b). Sequential Temporal GOLOG. Principles of Knowledge Representation and Reasoning: Proceedings of the Sixth International Conference (KR'98).

R. Rogers (1971). Mathematical logic and formalized theories - A Survey of Basic Concepts and Results. Amsterdam, London, North-Holland Publishing Company.

S. Russell und P. Norvig (1995). Artificial Intelligence- A modern Approach, Prentice Hall International, Inc.

D. Sanella (1988). A survey of formal software development methods. Edinburgh, Dept. of Artificial Intelligence and Dept. of Computer Science, University of Edinburgh.

L. K. Schubert (1990). Monotonic Solution of the Frame Problem in the Situation Calculus: an Efficient Method for Worlds with Fully Specified Actions. Knowledge Representation and Defeasible Reasoning. H.E. Kyberg, R. P. Loui und G. N. Carlson. Boston, Mass., Kluwer Academic Press: 23-67.

R. Waldinger (1977). Achieving several Goals simultaneously. Machine Intelligence 8. E. Block und D. Michie. Edinburgh, Ellis Horwood: 94-136.

Anhang A: GOLOG Interpreter

Der GOLOG Interpreter

```
/*               A GOLOG INTERPRETER IN ECLIPSE PROLOG

                           Feb. 6, 1998

This software was developed by the Cognitive Robotics Group under the
direction of Hector Levesque and Ray Reiter.

       Do not distribute without permission.
       Include this notice in any copy made.


        Copyright (c) 1992-1997 by The University of Toronto,
                       Toronto, Ontario, Canada.

                         All Rights Reserved

Permission to use, copy, and modify, this software and its
documentation for research purpose is hereby granted without fee,
provided that the above copyright notice appears in all copies and
that both the copyright notice and this permission notice appear in
supporting documentation, and that the name of The University of
Toronto not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior
permission.  The University of Toronto makes no representations about
the suitability of this software for any purpose.  It is provided "as
is" without express or implied warranty.

THE UNIVERSITY OF TORONTO DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS
SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS, IN NO EVENT SHALL THE UNIVERSITY OF TORONTO BE LIABLE FOR ANY
SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER
RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF
CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN
CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

*********************************************************************/

:- set_flag(print_depth,100).
:- nodbgcomp.
:- dynamic(proc/2).            /* Compiler directives. Be sure   */
:- set_flag(all_dynamic, on).  /* that you load this file first! */

:- op(800, xfy, [&]).   /* Conjunction */
:- op(850, xfy, [v]).   /* Disjunction */
:- op(870, xfy, [=>]).  /* Implication */
:- op(880,xfy, [<=>]).  /* Equivalence */
:- op(950, xfy, [:]).   /* Action sequence */
:- op(960, xfy, [#]).   /* Nondeterministic action choice */

do(E1 : E2,S,S1) :- do(E1,S,S2), do(E2,S2,S1).
do(?(P),S,S) :- holds(P,S).
do(E1 # E2,S,S1) :- do(E1,S,S1) ; do(E2,S,S1).
do(if(P,E1,E2),S,S1) :- do((?(P) : E1) # (?(-P) : E2),S,S1).
do(star(E),S,S1) :- S1 = S ; do(E : star(E),S,S1).
do(while(P,E),S,S1):- do(star(?(P) : E) : ?(-P),S,S1).
do(pi(V,E),S,S1) :- sub(V,_,E,E1), do(E1,S,S1).
do(E,S,S1) :- proc(E,E1), do(E1,S,S1).
do(E,S,do(E,S)) :- primitive_action(E), poss(E,S).

/* sub(Name,New,Term1,Term2): Term2 is Term1 with Name replaced by New. */

sub(X1,X2,T1,T2) :- var(T1), T2 = T1.
sub(X1,X2,T1,T2) :- not var(T1), T1 = X1, T2 = X2.
sub(X1,X2,T1,T2) :- not T1 = X1, T1 =..[F|L1], sub_list(X1,X2,L1,L2),
                    T2 =..[F|L2].
sub_list(X1,X2,[],[]).
sub_list(X1,X2,[T1|L1],[T2|L2]) :- sub(X1,X2,T1,T2), sub_list(X1,X2,L1,L2).

/* The holds predicate implements the revised Lloyd-Topor
   transformations on test conditions.  */

holds(P & Q,S) :- holds(P,S), holds(Q,S).
holds(P v Q,S) :- holds(P,S); holds(Q,S).
holds(P => Q,S) :- holds(-P v Q,S).
holds(P <=> Q,S) :- holds((P => Q) & (Q => P),S).
holds(-(-P),S) :- holds(P,S).
```

```
holds(-(P & Q),S) :- holds(-P v -Q,S).
holds(-(P v Q),S) :- holds(-P & -Q,S).
holds(-(P => Q),S) :- holds(-(-P v Q),S).
holds(-(P <=> Q),S) :- holds(-((P => Q) & (Q => P)),S).
holds(-all(V,P),S) :- holds(some(V,-P),S).
holds(-some(V,P),S) :- not holds(some(V,P),S).  /* Negation */
holds(-P,S) :- isAtom(P), not holds(P,S).     /* by failure */
holds(all(V,P),S) :- holds(-some(V,-P),S).
holds(some(V,P),S) :- sub(V,_,P,P1), holds(P1,S).

/* The following clause treats the holds predicate for non fluents, including
   Prolog system predicates. For this to work properly, the GOLOG programmer
   must provide, for all fluents, a clause giving the result of restoring
   situation arguments to situation-suppressed terms, for example:
         restoreSitArg(ontable(X),S,ontable(X,S)).            */

holds(A,S) :- restoreSitArg(A,S,F), F ;
              not restoreSitArg(A,S,F), isAtom(A), A.

isAtom(A) :- not (A = -W ; A = (W1 & W2) ; A = (W1 => W2) ;
    A = (W1 <=> W2) ; A = (W1 v W2) ; A = some(X,W) ; A = all(X,W)).

restoreSitArg(poss(A),S,poss(A,S)).
```

Der reaktive sequentielle temporale GOLOG Interpreter

```
:- set_flag(print_depth,100).
:- nodbgcomp.
:- dynamic(proc/2).            /* Compiler directives. Be sure   */
:- set_flag(all_dynamic, on).  /* that you load this file first! */

:- op(800, xfy, [&]).   /* Conjunction */
:- op(850, xfy, [v]).   /* Disjunction */
:- op(870, xfy, [=>]).  /* Implication */
:- op(880,xfy, [<=>]).  /* Equivalence */
:- op(950, xfy, [:]).   /* Action sequence */
:- op(960, xfy, [#]).   /* Nondeterministic action choice */


doR(E1 : E2,Rules,S,S1) :- doR(E1,Rules,S,S2), doR(E2,Rules,S2,S1).
doR(?(P),Rules,S,S) :- holds(P,S).
doR(E1 # E2,Rules,S,S1) :- doR(E1,Rules,S,S1) ; doR(E2,Rules,S,S1).
doR(if(P,E1,E2),Rules,S,S1) :- doR(?(P) : E1 # ?(-P) : E2,Rules,S,S1).
doR(star(E),Rules,S,S1) :- S1 = S ; doR(E : star(E),Rules,S,S1).
doR(while(P,E),Rules,S,S1):- doR(star(?(P) : E) : ?(-P),Rules,S,S1).
doR(pi(V,E),Rules,S,S1) :- sub(V,_,E,E1), doR(E1,Rules,S,S1).
doR(E,Rules,S,S1) :- proc(E,E1), doR(E1,Rules,S,S1).

doR(E,Rules,S,S1) :- primitive_action(E), poss(E,S), start(S,T1), time(E,T2), T1 $<= T2,
                     doR(Rules,Rules,do(E,S),S1).


/* sub(Name,New,Term1,Term2): Term2 is Term1 with Name replaced by New. */

sub(X1,X2,T1,T2) :- var(T1), T2 = T1.
sub(X1,X2,T1,T2) :- not var(T1), T1 = X1, T2 = X2.
sub(X1,X2,T1,T2) :- not T1 = X1, T1 =..[F|L1], sub_list(X1,X2,L1,L2),
                    T2 =..[F|L2].
sub_list(X1,X2,[],[]).
sub_list(X1,X2,[T1|L1],[T2|L2]) :- sub(X1,X2,T1,T2), sub_list(X1,X2,L1,L2).

/* The holds predicate implements the revised Lloyd-Topor
   transformations on test conditions.  */

holds(P & Q,S) :- holds(P,S), holds(Q,S).
holds(P v Q,S) :- holds(P,S); holds(Q,S).
holds(P => Q,S) :- holds(-P v Q,S).
holds(P <=> Q,S) :- holds((P => Q) & (Q => P),S).
holds(-(-P),S) :- holds(P,S).
holds(-(P & Q),S) :- holds(-P v -Q,S).
holds(-(P v Q),S) :- holds(-P & -Q,S).
holds(-(P => Q),S) :- holds(-(-P v Q),S).
holds(-(P <=> Q),S) :- holds(-((P => Q) & (Q => P)),S).
holds(-all(V,P),S) :- holds(some(V,-P),S).
holds(-some(V,P),S) :- not holds(some(V,P),S).  /* Negation */
holds(-P,S) :- isAtom(P), not holds(P,S).     /* by failure */
holds(all(V,P),S) :- holds(-some(V,-P),S).
holds(some(V,P),S) :- sub(V,_,P,P1), holds(P1,S).

/* The following clause treats the holds predicate for non fluents, including
```

```
   Prolog system predicates. For this to work properly, the GOLOG programmer
   must provide, for all fluents, a clause giving the result of restoring
   situation arguments to situation-suppressed terms, for example:
         restoreSitArg(ontable(X),S,ontable(X,S)).            */

holds(A,S) :- restoreSitArg(A,S,F), F ;
              not restoreSitArg(A,S,F), isAtom(A), A.

isAtom(A) :- not (A = -W ; A = (W1 & W2) ; A = (W1 => W2) ;
    A = (W1 <=> W2) ; A = (W1 v W2) ; A = some(X,W) ; A = all(X,W)).

restoreSitArg(poss(A),S,poss(A,S)).

/* Situationsargument wiederherstellen für start */
restoreSitArg(start(T),S,start(S,T)).

/* Prädikat start definieren */
start(do(A,S),T) :- time(A,T).
```

Anhang B:
Die Implementation eines Grundbuchsystems in Prolog

```
/*****************************************************************************
 *                                                                           *
 *                    Das Modell eines Grundbuchsystems                      *
 *                                                                           *
 *      vom 21.5.98 / Steffen Bittner                                        *
 *                                                                           *
 *****************************************************************************/

 % letzte Änderung am: 21.7.98


% Constraint Solving Library
:- lib(r).

:- compile(rstgolog).

%*****************************************************************************
%*************Implementation der materiellen Rechtsverhältnisse***************
%*****************************************************************************


%**************************************
% Primitive Action Declarations
%**************************************

primitive_action(verkaufe_Grundstueck(G,P1,P2,T)).
primitive_action(bestelle_Hypothek(G,P1,P2,W,T)).
primitive_action(zurueckzahlen_Hypothek(G,P1,P2,W,T)).
primitive_action(bestelle_Nieszbrauch(G,P1,P2,T)).
primitive_action(vereinige_Grundstuecke(G1,G2,P,T)).
primitive_action(teile_Grundstueck(G1,G2,P,FL,T)).
primitive_action(stirbt_Person(P,T)).


%**************************************
% Wiederherstellung der Situationsargumente
%**************************************

restoreSitArg(grundstueck(G),S,grundstueck(G,S)).
restoreSitArg(person(P),S,person(P,S)).
restoreSitArg(eigentuemer(G,P),S,eigentuemer(G,P,S)).
restoreSitArg(glaeubiger_Hypothek(G,P,W),S,glaeubiger_Hypothek(G,P,W,S)).
restoreSitArg(flurstueck(Fl),S,flurstueck(Fl,S)).
restoreSitArg(grundstueck_Flurstuecke_m(G,Fl),S,(G,Fl,S)).

%**************************************
% externe Transaktionen definieren
%**************************************

%alle Transaktionen des Modells der materiellen Rechtsverhältnisse sind externe
Transaktionen

extAction(verkaufe_Grundstueck(G,P1,P2,T)).
extAction(bestelle_Hypothek(G,P1,P2,W,T)).
extAction(zurueckzahlen_Hypothek(G,P1,P2,W,T)).
extAction(bestelle_Nieszbrauch(G,P1,P2,T)).
extAction(vereinige_Grundstuecke(G1,G2,P,T)).
extAction(teile_Grundstueck(G1,G2,P,FL,T)).
extAction(stirbt_Person(P,T)).


%**************************************
% Preconditions für primitive Transaktionen
%**************************************

poss(verkaufe_Grundstueck(G,P1,P2,T),S) :- grundstueck(G,S), person(P1,S),
       person(P2,S), eigentuemer(G,P1,S).

poss(bestelle_Hypothek(G,P1,P2,W,T),S) :- grundstueck(G,S), person(P1,S),
       person(P2,S), eigentuemer(G,P2,S), W$>0.

poss(zurueckzahlen_Hypothek(G,P1,P2,W,T),S) :- grundstueck(G,S), person(P1,S),
       person(P2,S), eigentuemer(G,P1,S),
      glaeubiger_Hypothek(G,P2,W,S).

poss(bestelle_Nieszbrauch(G,P1,P2,T),S) :- grundstueck(G,S), person(P1,S), person(P2,S),
```

```
        eigentuemer(G,P1,S).

poss(vereinige_Grundstuecke(G1,G2,P,T),S) :- grundstueck(G1,S),grundstueck(G2,S),
        person(P2,S),eigentuemer(G1,P,S), eigentuemer(G2,P,S).

poss(teile_Grundstueck(G1,G2,P,FL,T),S) :- grundstueck(G1,S), person(P,S),
        flurstueck(FL,S),eigentuemer(G1,P,S), grundstueck_Flurstuecke_m(G1,Fl,S),
        grundstueck_Flurstuecke_m(G1,Fl2,S), not Fl=Fl2, not grundstueck(G2,S).

poss(stirbt_Person(P,T),S) :- person(P,S).

%************************************
% Successor State Axiome
%************************************

person(P,do(A,S)) :- person(P,S), not a=stirbt_Person(P,S).

grundstueck(G,do(A,S)) :- A=teile_Grundstueck(_,G,_,_,T);
         grundstueck(G,S),
         not A=vereinige_Grundstuecke(_,G,_,T).

flurstueck(Fl,do(A,S)) :- flurstueck(Fl,S).

eigentuemer(G,P,do(A,S)) :- A=verkaufe_Grundstueck(G,_,P,T);A=teile_Grundstueck(_,G,P,_,T);
         eigentuemer(G,P,S),
         not A=verkaufe_Grundstueck(G,P,_,T), not A=vereinige_Grundstuecke(_,G,P,T).

glaeubiger_Hypothek(G,P,W,do(A,S)) :- A=bestelle_Hypothek(G,P,_,W,T);
        A=vereinige_Grundstuecke(G,G1,P1,T), glaeubiger_Hypothek(G1,P,W,S);
         glaeubiger_Hypothek(G,P,W,S),
        not A=zurueckzahlen_Hypothek(G,_,P,W,T),
        not A=vereinige_Grundstuecke(_,G,_,T).

nieszbraucher(G,P,do(A,S)) :- A=bestelle_Nieszbrauch(G,_,P,T);
        A=vereinige_Grundstuecke(G,G1,P1,T), nieszbraucher(G1,P,S);
        nieszbraucher(G,P,S),
        not A=stirbt_Person(P,T),
        not A=vereinige_Grundstuecke(_,G,P1,T).

grundstueck_Flurstuecke_m(G,Fl,do(A,S)) :- A=vereinige_Grundstuecke (G,G1,_,T),
        grundstueck_Flurstuecke_m(G1,Fl,S);
        A=teile_Grundstueck(G2,G,_,Fl,T);
        grundstueck_Flurstuecke_m(G,Fl,S),
        not A=teile_Grundstueck(G,G3,_,Fl,T),
        not A=vereinige_Grundstuecke(G4,G,_,T).


%************************************
% Die initiale Situation
%************************************

time(verkaufe_Grundstueck(G,P1,P2,T),T).
time(bestelle_Hypothek(G,P1,P2,W,T),T).
time(zurueckzahlen_Hypothek(G,P1,P2,W,T),T).
time(bestelle_Nieszbrauch(G,P1,P2,T),T).
time(vereinige_Grundstuecke(G1,G2,P,T),T).
time(teile_Grundstueck(G1,G2,P,FL,T),T).
time(stirbt_Person(P,T),T).

%***********Objekte*************************
person(p1,s0).
person(p2,s0).
person(p3,s0).
person(p4,s0).
grundstueck(g1,s0).
grundstueck(g2,s0).
grundstueck(g3,s0).
grundstueck(g4,s0).
flurstueck(f1,s0).
flurstueck(f2,s0).
flurstueck(f3,s0).
flurstueck(f4,s0).
flurstueck(f5,s0).
flurstueck(f6,s0).

%***********relationen***********************
grundstueck_Flurstuecke_m(g1,f1,s0).
grundstueck_Flurstuecke_m(g1,f2,s0).
grundstueck_Flurstuecke_m(g2,f3,s0).
grundstueck_Flurstuecke_m(g2,f4,s0).
grundstueck_Flurstuecke_m(g3,f5,s0).
grundstueck_Flurstuecke_m(g4,f6,s0).
```

```
eigentuemer(g1,p1,s0).
eigentuemer(g2,p2,s0).
eigentuemer(g3,p3,s0).
eigentuemer(g4,p1,s0).
glaeubiger_Hypothek(g1,p3,1000,s0).

%*************************************************************************
%*************************Implementation des Grundbuches******************
%*************************************************************************

%*************************************
% Primitive Action Declarations
%*************************************

% interne Transaktionen
primitive_action(eintragen_Vollrecht(G,P,R,T)).
primitive_action(loeschen_Vollrecht(G,P,R,T)).
primitive_action(eintragen_Teilrecht_Nutzung(G,P,R,T)).
primitive_action(loeschen_Teilrecht_Nutzung(G,P,R,T)).
primitive_action(eintragen_Teilrecht_Wert(G,P,R,W,T)).
primitive_action(loeschen_Teilrecht_Wert(G,P,R,W,T)).
primitive_action(eintragen_vereinigen_Grundstueck(G1,G2,T)).
primitive_action(eintragen_teilen_Grundstueck(G1,G2,Fl,T)).
primitive_action(start_Grundbuchverfahren(G,T)).
primitive_action(end_Grundbuchverfahren(G,T)).
primitive_action(loesche_Antrag(An,G1,G2,Fl,P1,P2,W,T)).

%externe Transaktionen
primitive_action(register_Dokument(G,P1,P2,R,T)).
primitive_action(register_Antrag(An,G1,G2,Fl,P1,P2,W,T)).

%*************************************
% Wiederherstellung der Situationsargumente
%*************************************

%Fluents, die das materielle Recht betreffen

restoreSitArg(eingetragenes_Recht(G,P,R),S,eingetragenes_Recht(G,P,R,S)).
restoreSitArg(grundstueck_Flurstuecke(G,Fl),S,grundstueck_Flurstuecke(G,Fl,S)).
restoreSitArg(wert_Recht(G,P,R,W),S,wert_Recht(G,P,R,W,S)).

%Fluents, die das formelle Recht betreffen

restoreSitArg(laufend_Grundbuchverfahren(G),S,laufend_Grundbuchverfahren(G,S)).
restoreSitArg(antrag(An,G1,G2,Fl,P1,P2,W),S,antrag(An,G1,G2,Fl,P1,P2,W,S)).
restoreSitArg(dokument(G,P1,P2,R),S,dokument(G,P1,P2,R,S)).

%***********************************
% externe Transaktionen definieren
%***********************************

extAction(register_Dokument(G,P1,P2,R,T)).
extAction(register_Antrag(An,G1,G2,Fl,P1,P2,W,T)).

%***********************************
% Preconditions für primitive Aktionen
%***********************************

poss(eintragen_Vollrecht(G,P,R,T),S) :- vollrecht(R), not eingetragenes_Recht(G,P,R,S).

poss(loeschen_Vollrecht(G,P,R,T),S) :- vollrecht(R), eingetragenes_Recht(G,P,R,S).

poss(eintragen_Teilrecht_Nutzung(G,P,R,T),S) :- teilrecht_Nutzung(R),
       not eingetragenes_Recht(G,P,R,S).

poss(loeschen_Teilrecht_Nutzung(G,P,R,T),S) :- teilrecht_Nutzung(R),
       eingetragenes_Recht(G,P,R,S).

poss(eintragen_Teilrecht_Wert(G,P,R,W,T),S) :- teilrecht_Wert(R),
       not eingetragenes_Recht(G,P,R,S), W$>0.

poss(loeschen_Teilrecht_Wert(G,P,R,W,T),S) :- teilrecht_Wert(R),
       eingetragenes_Recht(G,P,R,S), wert_Recht(G,P,R,W,S).

poss(eintragen_vereinigen_Grundstueck(G1,G2,T),S) :- eingetragenes_Recht(G1,P,R,S),
       eingetragenes_Recht(G2,P,R,S), vollrecht(R).

poss(eintragen_teilen_Grundstueck(G1,G2 ,Fl,T),S) :- eingetragenes_Recht(G1,P,R,S),
       vollrecht(R), grundstueck_Flurstuecke(G1,Fl,S),
       grundstueck_Flurstuecke(G1,Fl2,S), not Fl=Fl2.

poss(start_Grundbuchverfahren(G,T),S) :- not laufend_Grundbuchverfahren(G,S).
```

```
poss(end_Grundbuchverfahren(G,T),S) :- laufend_Grundbuchverfahren(G,S).

poss(register_Dokument(G,P1,P2,R,T),S) :- not dokument(G,P1,P2,R,S).

poss(register_Antrag(An,G1,G2,Fl,P1,P2,W,T),S) :-  antragsart(An).

poss(loesche_Antrag(An,G1,G2,Fl,P1,P2,W,T),S) :- antrag(An,G1,G2,Fl,P1,P2,W,S).


%*************************************
% Successor State Axioms
%*************************************

% Fluents, die formelles Recht betreffen

dokument(G,P1,P2,R,do(A,S)) :- A=register_Dokument(G,P1,P2,R,T); dokument(G,P1,P2,R,S).

antrag(An,G1,G2,Fl,P1,P2,W,do(A,S)) :- A = register_Antrag(An,G1,G2,Fl,P1,P2,W,T);
       antrag(An,G1,G2,Fl,P1,P2,W,S), not A = loesche_Antrag (An,G1,G2,Fl,P1,P2,W,T).

laufend_Grundbuchverfahren(G,do(A,S)) :- A=start_Grundbuchverfahren(G,T);
       laufend_Grundbuchverfahren(G,S),not A=end_Grundbuchverfahren(G,T).

% Fluents, die materielles Recht betreffen

eingetragenes_Recht(G,P,R,do(A,S)) :- A=eintragen_Vollrecht(G,P,R,T);
       A=eintragen_Teilrecht_Wert(G,P,R,W,T); A=eintragen_Teilrecht_Nutzung(G,P,R,T);
       eingetragenes_Recht(G,P,R,S),
       not A=loeschen_Vollrecht(G,P,R,T), not A=loeschen_Teilrecht_Wert(G,P,R,W,T),
       not A=loeschen_Teilrecht_Nutzung(G,P,R,T).

wert_Recht (G,P,R,W,do(A,S)) :- A=eintragen_Teilrecht_Wert(G,P,R,W,T);
       wert_Recht(G,P,R,W,S),
       not A=loeschen_Teilrecht_Wert(G,P,R,W,T).

grundstueck_Flurstuecke(G,Fl,do(A,S)) :- A=eintragen_vereinigen_Grundstueck (G,G1,T),
       grundstueck_Flurstuecke(G1,Fl,S); A=eintragen_teilen_Grundstueck(G2,G,Fl,T);
       grundstueck_Flurstuecke(G,Fl,S),
       not A=eintragen_teilen_Grundstueck(G,G3,Fl,T),
       not A=eintragen_vereinigen_Grundstuecke(G4,G,T).



%*************************************
% GOLOG Proceduren
%*************************************

% Das Grundbuchamt

proc(gba,
       while (-(halt),
              (if(-(some(an, some(g1, some(g2, some(fl, some(p1, some(p2, some(w,
                     antrag(an,g1,g2,fl,p1,p2,w)))))))),
              % then
                     pi(t, ?(now(t)) : wait(t)),
              % else

                 (pi (g1, pi(g2, pi(fl, pi(p1, pi(p2, pi(w,
                     (?(antrag(transfer_e,g1,g2,fl,p1,p2,w)):
                            gb_transfer_Eigentum(g1,p1,p2 ):
                            pi(t, ?(now(t)):
                            loesche_Antrag(transfer_e,g1,g2,fl,p1,p2,w,t))) #
                     (?(antrag(hypothek_e,g1,g2,fl,p1,p2,w)) :
                            gb_eintragen_Hypothek(g1,p1,w):
                            pi(t, ?(now(t)):
                            loesche_Antrag(hypothek_e,g1,g2,fl,p1,p2,w,t))) #
                     (?(antrag(hypothek_l,g1,g2,fl,p1,p2,w)):
                            gb_loeschen_Hypothek(g1,p1,w):
                            pi(t, ?(now(t)):
                            loesche_Antrag(hypothek_l,g1,g2,fl,p1,p2,w,t))) #
                     (?(antrag(nieszbrauch_e,g1,g2,fl,p1,p2,w)):
                            gb_eintragen_Nieszbrauch (g1,p1):
                            pi(t, ?(now(t)):
                            loesche_Antrag(nieszbrauch_e,g1,g2,fl,p1,p2,w,t))) #
                     (?(antrag(teilung_e,g1,g2,fl,p1,p2,w)): gb_Teilung (g1,g2,fl,p1):
                            pi(t, ?(now(t)):
                            loesche_Antrag(teilung_e,g1,g2,fl,p1,p2,w,t))) #
                     (?(antrag(vereinigung_e,g1,g2,fl,p1,p2,w)):
                            gb_Vereinigung (g1,g2,p1):
                            pi(t, ?(now(t)):
```

```
                              loesche_Antrag(vereinigung_e,g1,g2,fl,p1,p2,w,t)))
                  )))))))
              ))
       )
).


% Die Grundbuchverfahren

%Verfahren zur Eintragung eines Eigentumstransfers

proc( gb_transfer_Eigentum(G1,P1,P2 ),
    pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
    ?(eingetragenes_Recht(G1,P1,eigentum)):
    ?(dokument(G1,P1,P2,eigentum)):
    pi (t, (?(now(t))) : eintragen_Vollrecht(G1,P2,eigentum,t):
    loeschen_Vollrecht(G1,P1,eigentum,t): end_Grundbuchverfahren (G1,t))
).

%Verfahren zur Eintragung einer Hypothek

proc( gb_eintragen_Hypothek(G1,P1,W ),
        pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
        (?(some(p, (eingetragenes_Recht(G1,p,eigentum) & dokument(G1,p,P1,hypothek))))):
        pi (t, (?(now(t))) : eintragen_Teilrecht_Wert (G1,P1,hypothek,W,t):
               end_Grundbuchverfahren (G1,t))
).

%Verfahren zur Löschung einer Hypothek

proc( gb_loeschen_Hypothek(G1,P1,W),
        pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
        pi(p, pi (t, (?(now(t))) :
       (?(eingetragenes_Recht(G1,p,hypothek) & dokument(G1,p,P1,hypothek) &
          eingetragenes_Recht(G1,P1,eigentum) & wert_Recht(G1,p,hypothek,W))):
             loeschen_Teilrecht_Wert (G1,p,hypothek,W,t):
             end_Grundbuchverfahren (G1,t)))
).

%Verfahren zur Eintragung eines Niesbrauches

proc( gb_eintragen_Nieszbrauch (G1,P1),
        pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
        (?(some(p, (eingetragenes_Recht(G1,p,eigentum) & dokument(G1,p,P1,nieszbrauch))))):
        pi (t, (?(now(t))) : eintragen_Teilrecht_Nutzung (G1,P1,nieszbrauch,t):
               end_Grundbuchverfahren (G1,t))
).

%Verfahren zur Eintragung einer Grundstücksteilung

proc( gb_Teilung (G1,G2,Fl,P1),
        pi (t, (?(now(t))) : start_Grundbuchverfahren (G1,t)):
        (?(eingetragenes_Recht(G1,P1,eigentum))):
       (?(grundstueck_Flurstuecke(G1,Fl))):
        pi (t, (?(now(t))) : eintragen_teilen_Grundstueck (G1,G2,Fl,t):
               start_Grundbuchverfahren (G2,t): eintragen_Vollrecht(G2,P1,eigentum,t):
               end_Grundbuchverfahren (G1,t) : end_Grundbuchverfahren(G2,t))
).

%Verfahren zur Eintragung einer Grundstücksvereinigung

proc( gb_Vereinigung (G1,G2,P1),
       pi (t, (?(now(t))) :
       start_Grundbuchverfahren (G1,t): start_Grundbuchverfahren (G2,t):
        (?(eingetragenes_Recht(G1,P1,eigentum) & eingetragenes_Recht(G2,P1,eigentum))):
       eintragen_vereinigen_Grundstueck (G1,G2,t):
      while ( (some(p, some(r, (eingetragenes_Recht (G2,p,r))))),
                  pi(p, pi( r, (?(eingetragenes_Recht (G2,p,r) & vollrecht(r))) :
                    loeschen_Vollrecht(G2,p,r,t))) #
                  pi(p, pi( r, (?(eingetragenes_Recht (G2,p,r) & teilrecht_Nutzung(r))) :
                    loeschen_Teilrecht_Nutzung(G2,p,r,t):
                    eintragen_Teilrecht_Nutzung(G1,p,r,t))) #
                  pi(p, pi( r, pi(w, (?(eingetragenes_Recht (G2,p,r) & teilrecht_Wert(r) &
                                        wert_recht(G2,p,r,w ))) :
                    loeschen_Teilrecht_Wert(G2,p,r,w, t):
                    eintragen_Teilrecht_Wert(G1,p,r,w,t))))
        ):
        end_Grundbuchverfahren (G1,t): end_Grundbuchverfahren (G2,t))
).

%******Interrupts*********
```

```
proc(interrupts,
       ((?(some(g,some(p,some(w, beschwerde(g,p,eigentum,w,eintragen)
              & inkons_neE(g,p)))))) :
              (?(wln( "neues Vollrecht eintragen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,eigentum,w,eintragen) & inkons_neE(g,p))) :
                     (pi(t, (?(now(t))) : eintragen_Vollrecht(g,p,eigentum,t) :
                            loeschen_Beschwerde(g,p,eigentum,w,eintragen,t)) ))))

       ) #
       ((?(some(g,some(p,some(w, beschwerde(g,p,eigentum,w,loeschen)
              & inkons_eE(g,p)))))) :
              (?(wln( "unrichtiges Vollrecht loeschen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,eigentum,w,loeschen) & inkons_eE(g,p))) :
                     (pi(t, (?(now(t))) : loeschen_Vollrecht(g,p,eigentum,t) :
                            loeschen_Beschwerde(g,p,eigentum,w,loeschen,t)) ))))

       ) #
       ((?(some(g,some(p,some(w, beschwerde(g,p,hypothek,w,eintragen)
              & inkons_neH(g,p,w)))))) :
              (?(wln( "neue Hypothek eintragen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,hypothek,w,eintragen) & inkons_neH(g,p,w))) :
                     (pi(t, (?(now(t))) : eintragen_Teilrecht_Wert(g,p,hypothek,w,t) :
                            loeschen_Beschwerde(g,p,hypothek,w,eintragen,t)) ))))

       ) #
       ((?(some(g,some(p,some(w, beschwerde(g,p,hypothek,w,loeschen)
              & inkons_eH(g,p,w)))))) :
              (?(wln( "unrichtige Hypothek loeschen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,hypothek,w,loeschen) & inkons_eH(g,p,w))) :
                     (pi(t, (?(now(t))) : loeschen_Teilrecht_Wert(g,p,hypothek,w,t) :
                            loeschen_Beschwerde(g,p,hypothek,w,loeschen,t)) ))))

       ) #
       ((?(some(g,some(p,some(w, beschwerde(g,p,nieszbrauch,w,eintragen)
              & inkons_neN(g,p)))))) :
              (?(wln( "neuen Nieszbrauch eintragen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,nieszbrauch,w,eintragen) & inkons_neN(g,p))) :
                     (pi(t, (?(now(t))) : eintragen_Teilrecht_Nutzung(g,p,nieszbrauch,t) :
                            loeschen_Beschwerde(g,p,nieszbrauch,w,eintragen,t)) ))))

       ) #
       ((?(some(g,some(p,some(w, beschwerde(g,p,nieszbrauch,w,loeschen)
              & inkons_eN(g,p)))))) :
              (?(wln( "unrichtigen Nieszbrauch loeschen"))):
              pi(g, pi(p, pi(w,
                     (?(beschwerde(g,p,nieszbrauch,w,loeschen) & inkons_eN(g,p))) :
                     (pi(t, (?(now(t))) : loeschen_Teilrecht_Nutzung(g,p,nieszbrauch,t) :
                            loeschen_Beschwerde(g,p,nieszbrauch,w,loeschen,t)) ))))

       ) #
       (?(-(some(g,some(p,some(w, beschwerde(g,p,eigentum,w,eintragen)
              & inkons_neE(g,p)))) v
            some(g,some(p,some(w, beschwerde(g,p,eigentum,w,loeschen)
              & inkons_eE(g,p)))) v
            some(g,some(p,some(w, beschwerde(g,p,hypothek,w,eintragen)
              & inkons_neH(g,p,w)))) v
            some(g,some(p,some(w, beschwerde(g,p,hypothek,w,loeschen)
              & inkons_eH(g,p,w)))) v
            some(g,some(p,some(w, beschwerde(g,p,nieszbrauch,w,eintragen)
              & inkons_neN(g,p)))) v
            some(g,some(p,some(w, beschwerde(g,p,nieszbrauch,w,loeschen)
              & inkons_eN(g,p)))))

       ))
).

% **********Schnittstelle für externe Transaktionen*******

proc(generateExogenousAction,
       pi(e, ?(requestExogenousAction(e)) :
       if(e = nil, ?(true), e))).

requestExogenousAction(E,S) :-
       write('Enter an exogenous action, or nil.'), read(E1),
/* IF exogenous action is nil, or is possible and externe Aktion THEN no problem */
              ((E1 = nil ; poss(E1,S), extAction(E1)) -> E = E1 ;
/* ELSE print error message, and try again. */
```

```
       write('?? Action not possible or not exogenous Action. Try again.'), nl,
       requestExogenousAction(E,S)).

restoreSitArg(requestExogenousAction(E),S,requestExogenousAction(E,S)).

% ************rules***************

proc(rules, interrupts : generateExogenousAction).

%*************************************
% Die initiale Situation
%*************************************

% eintragsfähige Rechte im Grundbuch

vollrecht(eigentum).
teilrecht_Wert(hypothek).
teilrecht_Nutzung(nieszbrauch).

%gültige Antragsarten
antragsart(transfer_e).
antragsart(hypothek_e).
antragsart(hypothek_l).
antragsart(nieszbrauch_e).
antragsart(teilung_e).
antragsart(vereinigung_e).

time(eintragen_Vollrecht(G,P,R,T),T).
time(loeschen_Vollrecht(G,P,R,T),T).
time(eintragen_Teilrecht_Nutzung(G,P,R,T),T).
time(loeschen_Teilrecht_Nutzung(G,P,R,T),T).
time(eintragen_Teilrecht_Wert(G,P,R,W,T),T).
time(loeschen_Teilrecht_Wert(G,P,R,W,T),T).
time(eintragen_vereinigen_Grundstueck(G1,G2,T),T).
time(eintragen_teilen_Grundstueck(G1,G2,Fl,T),T).
time(start_Grundbuchverfahren(G,T),T).
time(end_Grundbuchverfahren(G,T),T).

%externe Transaktionen
time(register_Dokument(G,P1,P2,R,T),T).
time(register_Antrag(An,G1,G2,Fl,P1,P2,W,T),T).
time(loesche_Antrag(An,G1,G2,Fl,P1,P2,W,T),T).

%Relationen
eingetragenes_Recht(g1,p1,eigentum,s0).
eingetragenes_Recht(g2,p2,eigentum,s0).
eingetragenes_Recht(g3,p3,eigentum,s0).
eingetragenes_Recht(g4,p1,eigentum,s0).
eingetragenes_Recht(g1,p3,hypothek,s0).
wert_Recht(g1,p3,hypothek,1000,s0).
grundstueck_Flurstuecke(g1,f1,s0).
grundstueck_Flurstuecke(g1,f2,s0).
grundstueck_Flurstuecke(g2,f3,s0).
grundstueck_Flurstuecke(g2,f4,s0).
grundstueck_Flurstuecke(g3,f5,s0).
grundstueck_Flurstuecke(g4,f6,s0).

%*****************************************************************************************
%                  Das Verhältnis von materiellem Recht und Grundbuch-
%                            die Markierung von Inkonsistenzen
%*****************************************************************************************

% inkonsistente Eigentumsverhältnisse
% vor Anwendung der Transformationsregeln
% inkonsE(G,P,S) <==> not( eigentuemer(G,P,S) <==> eingetragenes_Recht(G,P,eigentum,S)

% nach Anwendung der Transformationsregeln

inkonsE(G,P,S) :- eigentuemer(G,P,S), not eingetragenes_Recht(G,P,eigentum,S);
       eingetragenes_Recht(G,P,eigentum,S), not eigentuemer(G,P,S).

restoreSitArg(inkonsE(G,P),S,inkonsE(G,P,S)).

%ein Eigentumsrecht ist eingetragen, das materiellrechtlich nicht besteht
inkons_eE(G,P,S) :- eingetragenes_Recht(G,P,eigentum,S), not eigentuemer(G,P,S).

restoreSitArg(inkons_eE(G,P),S,inkons_eE(G,P,S)).

%ein Recht, das materiellrechtlich besteht, ist nicht eingetragen
inkons_neE(G,P,S) :- eigentuemer(G,P,S), not eingetragenes_Recht(G,P,eigentum,S).

restoreSitArg(inkons_neE(G,P),S,inkons_neE(G,P,S)).
```

```
% inkonsistente Eintragungen über eine Hypothek

inkonsH(G,P,W,S) :- glaeubiger_Hypothek(G,P,W,S), not eingetragenes_Recht(G,P,hypothek,S);
       glaeubiger_Hypothek(G,P,W,S), not wert_Recht(G,P,hypothek,W,S);
       eingetragenes_Recht(G,P,hypothek,S),wert_Recht(G,P,hypothek,W,S),
       not glaeubiger_Hypothek(G,P,W,S).

restoreSitArg(inkonsH(G,P,W),S,inkonsH(G,P,W,S)).

inkons_eH(G,P,W,S) :- eingetragenes_Recht(G,P,hypothek,S),wert_Recht(G,P,hypothek,W,S),
       not glaeubiger_Hypothek(G,P,W,S).

restoreSitArg(inkons_eH(G,P,W),S,inkons_eH(G,P,W,S)).

inkons_neH(G,P,W,S) :- glaeubiger_Hypothek(G,P,W,S),
       not eingetragenes_Recht(G,P,hypothek,S);
       glaeubiger_Hypothek(G,P,W,S), not wert_Recht(G,P,hypothek,W,S).

restoreSitArg(inkons_neH(G,P,W),S,inkons_neH(G,P,W,S)).

% inkonsistente Eintragungen über einen Nieszbrauch

inkonsN(G,P,S) :- nieszbraucher(G,P,S), not eingetragenes_Recht(G,P,nieszbrauch,S);
       eingetragenes_Recht(G,P,nieszbrauch,S), not nieszbraucher(G,P,S).

restoreSitArg(inkonsN(G,P),S,inkonsN(G,P,S)).

inkons_eN(G,P,S) :- eingetragenes_Recht(G,P,nieszbrauch,S), not nieszbraucher(G,P,S).

restoreSitArg(inkons_eN(G,P),S,inkons_eN(G,P,S)).

inkons_neN(G,P,S) :- nieszbraucher(G,P,S), not eingetragenes_Recht(G,P,nieszbrauch,S).

restoreSitArg(inkons_neN(G,P),S,inkons_neN(G,P,S)).

% inkonsistente Eintragungen über die Zuordnung der Grund- und Flurstücke

inkonsGF(G,Fl,S) :- grundstueck_Flurstuecke_m(G,Fl,S), not grundstueck_Flurstuecke(G,Fl,S);
       grundstueck_Flurstuecke(G,Fl,S), not grundstueck_Flurstuecke_m(G,Fl,S).

restoreSitArg(inkonsGF(G,Fl),S,inkonsGF(G,Fl,S)).

inkons_eGF(G,Fl,S) :- grundstueck_Flurstuecke(G,Fl,S),
       not grundstueck_Flurstuecke_m(G,Fl,S).

restoreSitArg(inkons_eGF(G,Fl),S,inkons_eGF(G,Fl,S)).

inkons_neGF(G,Fl,S) :- grundstueck_Flurstuecke_m(G,Fl,S),
       not grundstueck_Flurstuecke(G,Fl,S).

restoreSitArg(inkons_neGF(G,Fl),S,inkons_neGF(G,Fl,S)).


%**************Die Korrektur des unrichtigen Grundbuches*******************

beschwerde(G,P,R,W,Art,do(A,S)) :- A=register_Beschwerde(G,P,R,W,Art,T);
       beschwerde(G,P,R,W,Art,S), not A=loeschen_Beschwerde(G,P,R,W,Art,T).

primitive_action(register_Beschwerde(G,P,R,W,Art,T)).
primitive_action(loeschen_Beschwerde(G,P,R,W,Art,T)).
extAction(register_Beschwerde(G,P,R,W,Art,T)).

time(register_Beschwerde(G,P,R,W,Art,T),T).
time(loeschen_Beschwerde(G,P,R,W,Art,T),T).

poss(register_Beschwerde(G,P,R,W,Art,T),S):- not laufend_Grundbuchverfahren(G,S).
poss(loeschen_Beschwerde(G,P,R,W,Art,T),S) :- beschwerde(G,P,R,W,Art,S).
restoreSitArg(beschwerde(G,P,R,W,Art),S,beschwerde(G,P,R,W,Art,S)).

%*************Tools********************

wln(Text) :- write(Text), nl.

primitive_action(wait(T)).

poss(wait(T),S).
time(wait(T),T).

%stop ist die Transaktion, die das Programm beendet
poss(stop(T),S).
```

```
halt(do(A,S)) :- A=stop(T); halt(S).
restoreSitArg(halt,S,halt(S)).

time(stop(T),T).
primitive_action(stop(T)).
extAction(stop(T)).

start(s0,1).

%*************************************
% Fix a solution to the temporal constraints
%*************************************

chooseTimes(s0).
chooseTimes(do(A,S)) :- chooseTimes(S), time(A,T), rmin(T).

%*************************************
% "now" is a synonym for "start"
%*************************************

now(S,T) :- start(S,T).
restoreSitArg(now(T),S,now(S,T)).

%*************************************
% Utilities
%*************************************

prettyPrintSituation(S) :- makeActionList(S,Alist), nl,
                write(Alist), nl.

makeActionList(s0,[]).
makeActionList(do(A,S),L) :- makeActionList(S,L1), append(L1,[A],L).

% ******Anzeigen des Inhaltes der Datenbank******

makeGrundstuecksList(List,S) :- findall (G,grundstueck(G,S),List).
makePersonenList(List,S) :- findall (P,person(P,S),List).
makeGFMList(List,S) :- findall ((G,F), grundstueck_Flurstuecke_m(G,F,S), List).
makeEigentuemerList(List,S) :- findall ((G,P), eigentuemer(G,P,S), List).
makeGlaeubigerList(List,S) :- findall ((G,P,W), glaeubiger_Hypothek(G,P,W,S), List).
makeNieszbraucherList(List,S) :- findall ((G,P), nieszbraucher(G,P,S), List).
makeRechteList(List,S) :- findall ((G,P,R), eingetragenes_Recht(G,P,R,S), List).
makeWerteList(List,S) :- findall ((G,P,R,W), wert_Recht(G,P,R,W,S), List).
makeGFList(List,S) :- findall ((G,F), grundstueck_Flurstuecke(G,F,S), List).
makeDokuList (List,S) :- findall ((G,P1,P2,R), dokument(G,P1,P2,R,S), List).
makeInkonsEList(List,S) :- findall ((G,P), inkonsE(G,P,S), List).
makeInkons_eEList(List,S) :- findall ((G,P), inkons_eE(G,P,S), List).
makeInkons_neEList(List,S) :- findall ((G,P), inkons_neE(G,P,S), List).
makeInkonsHList(List,S) :- findall ((G,P,W), inkonsH(G,P,W,S), List).
makeInkons_eHList(List,S) :- findall ((G,P,W), inkons_eH(G,P,W,S), List).
makeInkons_neHList(List,S) :- findall ((G,P,W), inkons_neH(G,P,W,S), List).
makeInkonsNList(List,S) :- findall ((G,P), inkonsN(G,P,S), List).
makeInkons_eNList(List,S) :- findall ((G,P), inkons_eN(G,P,S), List).
makeInkons_neNList(List,S) :- findall ((G,P), inkons_neN(G,P,S), List).
makeInkonsGFList(List,S) :- findall ((G,Fl), inkonsGF(G,Fl,S), List).
makeInkons_eGFList(List,S) :- findall ((G,Fl), inkons_eGF(G,Fl,S), List).
makeInkons_neGFList(List,S) :- findall ((G,Fl), inkons_neGF(G,Fl,S), List).

printGrundstuecksList(S) :- write ("Grundstuecke: "), makeGrundstuecksList(List,S),
       write(List), nl.

printPersonenList(S) :- write ("Personen: "), makePersonenList(List,S), write(List), nl.

printGFMList(S) :- write ("Grundstücke/Flustuecke/m: "), makeGFMList(List,S), write(List),
       nl.

printEigentuemerList(S) :- write ("Eigentuemer: "), makeEigentuemerList(List,S),
       write(List), nl.

printGlaeubigerList(S) :- write ("Glaeubiger von Hypotheken: "),
       makeGlaeubigerList(List,S), write(List), nl.

printNieszbraucherList(S) :- write ("Nieszbraucher: "), makeNieszbraucherList(List,S),
       write(List), nl.

printRechteList(S) :- write ("eingetragene Rechte: "), makeRechteList(List,S), write(List),
       nl.

printWerteList(S) :- write ("Werte von Rechten: "), makeWerteList(List,S), write(List), nl.

printGFList(S) :- write ("Grundstücke/Flurstuecke: "), makeGFList(List,S), write(List), nl.
```

```
printDokuList(S) :- write ("Dokumente: "), makeDokuList(List,S), write(List), nl.

printInkonsEList(S) :- write ("inkonsistente Eigentumsverhältnisse: "),
       makeInkonsEList(List,S), write(List), nl.

printInkons_eEList(S) :- write ("   Einträge unrichtg: "), makeInkons_eEList(List,S),
       write(List), nl.

printInkons_neEList(S) :- write ("   Einträge nicht vorhanden: "),
       makeInkons_neEList(List,S), write(List), nl.

printInkonsHList(S) :- write ("inkonsistente Hypotheken: "), makeInkonsHList(List,S),
       write(List), nl.

printInkons_eHList(S) :- write ("   Einträge unrichtg: "), makeInkons_eHList(List,S),
       write(List), nl.

printInkons_neHList(S) :- write ("   Einträge nicht vorhanden: "),
       makeInkons_neHList(List,S), write(List), nl.

printInkonsNList(S) :- write ("inkonsistente Nieszbraeuche: "), makeInkonsNList(List,S),
       write(List), nl.

printInkons_eNList(S) :- write ("   Einträge unrichtg: "), makeInkons_eNList(List,S),
       write(List), nl.

printInkons_neNList(S) :- write ("   Einträge nicht vorhanden: "),
       makeInkons_neNList(List,S), write(List), nl.

printInkonsGFList(S) :- write ("inkonsistente Grundstuecke/Flurstuecke: "),
       makeInkonsGFList(List,S), write(List), nl.
printInkons_eGFList(S) :- write ("   Einträge unrichtg: "), makeInkons_eGFList(List,S),
       write(List), nl.
printInkons_neGFList(S) :- write ("   Einträge nicht vorhanden: "),
       makeInkons_neGFList(List,S), write(List), nl.

printFluentList(S) :- nl, wln("materielles Recht:"),
       printGrundstuecksList(S), printPersonenList(S),
       printGFMList(S),printEigentuemerList(S),
       printGlaeubigerList(S),printNieszbraucherList(S),
       nl, wln ("Grundbuch:"),
       printRechteList(S), printWerteList(S), printGFList(S),
       printDokuList(S),
       nl, wln ("Inkonsistenzen:"),
       printInkonsEList(S),printInkons_eEList(S),printInkons_neEList(S),
       printInkonsHList(S),printInkons_eHList(S),printInkons_neHList(S),
       printInkonsNList(S),printInkons_eNList(S),printInkons_neNList(S),
       printInkonsGFList(S),printInkons_eGFList(S),printInkons_neGFList(S).

%************************************
% Hauptprogramm
%************************************

gamt :- doR(gba,rules,s0,S), chooseTimes(S), prettyPrintSituation(S),printFluentList(S).
```