Master Thesis

im Rahmen des Universitätslehrganges „Geographical Information Science & Systems“ (UNIGIS MSc) am Zentrum für GeoInformatik (Z_GIS) der Paris Lodron-Universität Salzburg

zum Thema

Entwicklung der technischen Grundlagen eines GIS-basierten Grundeigentums-Informationssystems für Tansania

vorgelegt von

Dipl. Geograph Peter Schär

U1257, UNIGIS MSc Jahrgang 2006

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)“

Gutachter:
Ao. Univ. Prof. Dr. Josef Strobl

Biel, 28. April 2008

Danksagung

Ein besonderer Dank gebührt Dr. Martin Huber, dem Betreuer dieser Arbeit, der mich während der gesamten Master Thesis vom Themenvorschlag bis zum Verfassen des Schlusstextes jederzeit tatkräftig unterstützt hat.

Ebenfalls danken möchte ich Klaus Mithöfer. Das gemeinsame Thema hat die Arbeit an dieser Master Thesis sehr erleichtert. Die Diskussionen mit ihm – auch über grosse räumliche Distanz hinweg – waren immer fruchtbar und sehr hilfreich.

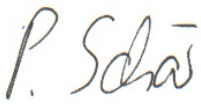
Gleichzeitig möchte ich auch meinen Arbeitgebern und Arbeitskollegen aus dem Bundesamt für Umwelt (BAFU) und dem Amt für Geoinformation des Kantons Bern (AGI) danken, die mir während des gesamten UNIGIS-Lehrganges ein hohes Mass an Arbeitszeitflexibilität ermöglicht haben.

Peter Schär
Biel, im April 2008

Eigenständigkeitserklärung

Ich versichere, diese Master Thesis ohne fremde Hilfe und ohne Verwendung anderer als der angeführten Quellen angefertigt zu haben, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäss übernommen wurden, sind als solche gekennzeichnet.

Biel, 28. April 2008



Peter Schär

Zusammenfassung

Tansania verfügt momentan über kein funktionierendes System zur einheitlichen Registrierung von Landansprüchen. Ansprüche im Wert von 29 Milliarden US-Dollar werden deshalb in einem extralegalen System verwaltet, das äusserst heterogen ist und die wirtschaftliche Entwicklung des Landes stark hemmt. Zur Behebung dieses Missstandes hat Hernando de Soto zusammen mit seinem Institute for Liberty and Democracy verschiedene Reformen vorgeschlagen (DE SOTO 2006), die auch von der tansansichen Regierung aufgenommen wurden. Eine dieser Reformen beschäftigt sich mit der Formalisierung und Legalisierung der extralegalen Landregister. Die vorliegende Arbeit beschreibt zusammen mit der Arbeit von MITHÖFER (2008) die formalen, organisatorischen und technischen Grundlagen eines sicheren und transparenten Frameworks zur Registrierung von Landansprüchen – genannt Universal Land Registry (ULR). Ganz speziell beschäftigt sich diese Arbeit mit den technischen Anforderungen, die an ein solches Framework gestellt werden und schlägt ein technisches Design vor, mit dem diese Anforderungen umgesetzt werden. Dieses Design wird in einem einfachen Prototypen basierend auf Open Source-Software praktisch umgesetzt.

Stichworte: Universal Land Registry, Tansania, Standards, Open Source

Abstract

Currently Tanzania does not dispose of a working system for registering land right information. 29 billion US$ worth of assets are therefore managed in an extralegal system, which is very heterogeneous and is hindering the economic development. To correct this flaw Hernando de Soto and his Institute for Liberty and Democracy proposed several reforms (DE SOTO 2006), that are being incorporated by the Government of Tanzania. One of these reforms takes care of the formalisation and legalisation of those extralegals land registries. The present master thesis together with a master thesis by MITHÖFER (2008) describes the formal, organizational and technical basics of a secure and transparent framework for registering land rights – called Universal Land Registry (ULR). This master thesis especially deals with the technical requirements to be meet by such a framework and proposes a technical design for implementing those requirements. The design is implemented as a simple prototype based on Open Source-Software.

Keywords: Universal Land Registry, Tanzania, Standards, Open Source

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AAA	Authentifizierung, Autorisierung und Abrechnung
API	Application Programming Interface
CCDM	Core Cadastral Domain Model
DBMS	Datenbank-Management-System
ERD	Entity-Relationship-Diagramm
FE	Filter Encoding
FIG	Fédération Internationale des Géomètres
GeoDRM	Geospatial Digital Rights Management
GeoXACML	Geospatial eXtensible Access Control Markup Language
GIS	Geographisches Informationssystem
GML	Geography Markup Language
GPS	Global Positioning System
GSDI	Global Spatial Data Infrastructure
GPX	GPS Exchange Format
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol over Secure Socket Layer (SSL)
ILD	Institute for Liberty and Democracy
JDBC	Java Database Connectivity
LDAP	Lightweight Directory Access Protocol
NGDI	Nationale Geodateninfrastruktur
ODBC	Open Database Connectivity
OGC	Open Geospatial Consortium
OMG	Object Management Group
PBFP	Property and Business Formalisation Programme
RDBMS	Relationales Datenbankmanagementsystem
RRR	Right, Restriction, Responsibility
SFS	Simple Feature Access (SQL option)
STDM	Social Tenure Domain Model
SSL	Secure Socket Layer
ULR	Universal Land Registry
UML	Unified Modeling Language
VPN	Virtual Private Network
WAS	Web Authentication Service
WFS	Web Feature Service

WMS	Web Map Service
WPOS/XCPF	Web Pricing & Ordering Service / XML Configuration & Pricing Format
WSS	Web Security Service
XML	Extensible Markup Language

1 Einführung

1.1 Problemstellung und Motivation

Das Recht auf Eigentum ist ein wichtiger Grundpfeiler jedes funktionierenden Staates und seines Wirtschaftssystems. Nicht zuletzt deshalb ist dieses Recht auch in die Allgemeinen Deklaration der Menschenrechte aufgenommen worden:

> "**Artikel 17**
>
> 1. Jeder hat das Recht, sowohl allein als auch in Gemeinschaft mit anderen Eigentum innezuhaben.
> 2. Niemand darf willkürlich seines Eigentums beraubt werden." (VEREINTE NATIONEN 1948)

Dabei ist der Staat in der Pflicht, die notwendigen gesetzlichen Grundlagen, Verfahren und Prozesse zu definieren und der Bevölkerung und der Wirtschaft zur Verfügung zu stellen. In Tansania ist der Staat nicht in der Lage, das gesetzlich verankerte Recht auf Grundeigentum zu garantieren und durchzusetzen, da kein funktionierendes staatliches System zur Registrierung und Erfassung von Grundeigentum existiert. Im Gegenteil: Wie eine Studie des Institute for Liberty and Democracy (ILD) unter der Leitung von Hernando de Soto gezeigt hat (DE SOTO 2006), haben sich verschiedene extralegale Prozeduren und Strukturen herausgebildet, die es der Bevölkerung ermöglichen, ihr Grundeigentum informell zu registrieren. In diesem extralegalen System haben sich mittlerweile Vermögenswerte von ca. 29 Milliarden US-Dollar angesammelt, ein erheblicher Anteil an der gesamten tansanischen Wirtschaft (DE SOTO 2006:21). Dieses extralegale System ist nicht etwa anarchisch aufgebaut, sondern basiert auf einem sich selbst-organisierenden System von Institutionen und Dokumenten, mit denen die Individuen ihre Belange regeln (DE SOTO 2006:26). Es gibt gute Gründe, weshalb das legale System nicht funktioniert und die Bevölkerung ihre Wirtschaftstätigkeit in den extralegalen Bereich verschoben hat. Das legale System ist für die ganz grosse Mehrheit der Bevölkerung (gegen 90%) schlicht und einfach nicht zugänglich (DE SOTO 2006:50). Folgende Faktoren sind dafür verantwortlich:

- Die offiziellen Prozeduren sind sehr teuer, es werden hohe Gebühren erhoben.
- Die Dauer für die Abwicklung einer offiziellen Prozedur ist extrem hoch.
- Die Amtssprache ist Englisch. Eine Mehrheit der Bevölkerung spricht aber nur Suaheli.

- Prozeduren können nur in wenigen Städten abgewickelt werden. Da der Bürger persönlich vorsprechen muss, ist dies für viele Landbewohner ein unüberwindbares Hindernis.

Es gilt festzuhalten, dass das extralegale System an und für sich recht gut funktioniert. Es gibt jedoch gewisse Probleme. Grundsätzlich ist es kein idealer Zustand, wenn ein grosser Teil der Wirtschaft extralegal funktionieren muss. Ausserdem ist das extralegale System sehr zersplittert und heterogen. Es mag innerhalb eines Dorfes oder eines Quartiers sehr gut funktionieren, die Informationen können aber nur schlecht überregional oder gar national verwendet werden. Ein Dokument, das Grundeigentum belegt, ist daher nur in einem eng begrenzten geographischen Raum gültig. Solche Dokumente können auch nicht als Sicherheiten für Kredite oder Hypotheken hinterlegt werden, was die wirtschaftliche Entwicklung entscheidend hemmt. Es ist daher notwendig, mit einer Reihe von Reformen das extralegale und das legale System zusammenzuführen. DE SOTO (2006:56-58) beschreibt eine ganze Reihe solcher Reformen. Eine dieser Reformen im Bereich Grundeigentum beinhaltet auch eine GIS-Komponente:

> "Improving the existing formalization procedure, possibly with, among other things, a simplified, standardized and low-cost formalization procedure that allows costly and time-consuming official surveys to be conducted by private professionals; a simplified, decentralized, massive, and low cost registration procedure with a geographical database." (DE SOTO 2006:56)

De Sotos Reformvorschläge wurden von der tansanischen Regierung mit Interesse entgegengenommen und das Property and Business Formalisation Programme initiiert (CLARKE ET AL. 2007:3). Dieses Programm firmiert auch unter seiner Suaheli-Bezeichnung MKURABITA. De Sotos Forderung nach einer einfachen, dezentralen und billigen Registrierungsprozedur wurde aufgegriffen und als Ziel verankert:

> "To set up a unified and decentralized registry system, that secures legal warranties of registered rights, is easily accessible, operates coordinated with the existing land registries [...] and is provided with simplified and low-cost procedures that facilitate and thereby encourage not only first registration of titles but also registration of subsequent transactions." (NPA 2006:9)

Dieses Themas hat sich eine Gruppe von Dozenten und Studierenden verschiedener Universitäten angenommen und ein allgemeines Landregister – nachfolgend Universal Land Registry (ULR) genannt – entworfen, das die obigen Anforderungen erfüllen soll (HUBER ET AL. 2008). Die parallel zur vorliegenden Arbeit verfasste Master Thesis von Klaus Mithöfer (MITHÖFER 2008) entwirft das organisatorische und formale Framework für eine ULR, d.h. Prozesse und Datenmodelle. Um die Anforderungen und Grundlagen für eine technische Umsetzung kümmert sich die vorliegende Arbeit.

1.2 Hypothesen

Aus den obengenannten Rahmenbedingungen lassen sich folgende Hypothesen ableiten:

1. Die Universal Land Registry – wie von MITHÖFER (2008) beschrieben – ist mit existierenden IT- und GIS-Mitteln praktisch umsetzbar.

2. Der Einsatz von Open Source-Komponenten erlaubt eine Umsetzung der ULR, die den relevanten IT- und GIS-Standards folgt.

3. Die Abstützung auf IT- und GIS-Standards garantiert, dass die ULR langfristig eingesetzt werden und als Kern einer künftigen nationalen Geodateninfrastruktur Tansanias dienen kann.

1.3 Ziele

Das Ziel dieser Arbeit besteht darin, ein technisches Design für die Umsetzung der Universal Land Registry zu entwickeln. Zu berücksichtigen sind dabei die Anforderungen aus den übergeordneten Arbeiten und Konzepten (MITHÖFER 2008, HUBER ET AL. 2008, NPA 2006) sowie die relevanten IT- und GIS-Standards. Abzuhandelnde Themen sind hierbei Anwendungsdesign, Systemarchitektur, Datenhaltung, Datenbereitstellung und Georeferenzierung.

Die Ergebnisse der Arbeit sollen anhand einer einfachen prototypischen Umsetzung (Aufbau einer ULR-Applikation) überprüft werden, um das Design einem ersten praktischen Test auszusetzen.

1.4 Lösungsansatz und Struktur

Die obigen Ziele sollen mit der in Abbildung 1.1 gezeigten Vorgehensweise erreicht werden, welche auch der formalen Struktur dieser Master Thesis entspricht.

Durch die Literaturrecherche (Kapitel 2) werden zunächst die technischen Grundlagen für eine ULR-Umsetzung ermittelt. Dabei sollen die relevanten technischen und methodischen Konzepte beschrieben werden sowie die Rahmenbedingungen, die aus den übergeordneten ULR-Konzepten und -Arbeiten entstehen. Daneben werden bereits bestehende Arbeiten und Forschungsergebnisse in diesem Bereich diskutiert. Die durch die Literaturrecherche ermittelten konkreten Anforderungen werden in einem separaten Kapitel (Kapitel 3) ausformuliert und festgehalten.

Anschliessend wird basierend auf den Anforderungen und den Ergebnissen der Literaturrecherche die eigentliche Umsetzung – die ULR-Applikation – entworfen und beschrieben. Dies geschieht zunächst durch die Identifizierung und Beschreibung aller Anwendungsfälle (Use Cases) in Kapitel 4 und danach mit dem Entwurf einer logischen Systemarchitektur in Kapitel 5. Dieses Design wird wie in den Zielen erwähnt auch in Form eines einfachen Prototypen umgesetzt, um es praktisch überprüfen zu können. Dies ist in Kapitel 7 ausgeführt. Im abschliessenden Kapitel ?? werden die Ergebnisse in einer Synthese diskutiert und ein Ausblick in die Zukunft geworfen.

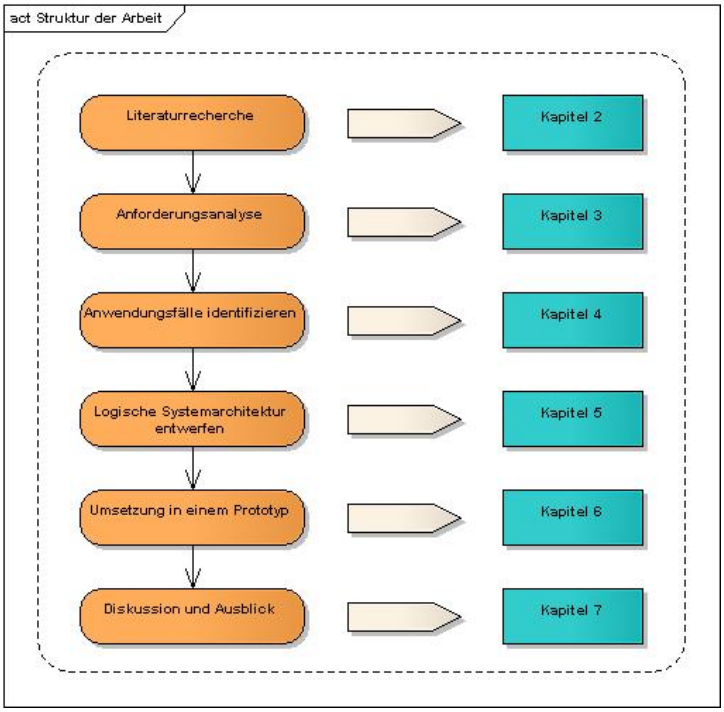


Abb. 1.1: Aufbau und Struktur der Master Thesis

1.5 Abgrenzung und Publikum

Es ist nicht Ziel dieser Arbeit, die theoretischen und konzeptionellen Grundlagen der ULR darzulegen. Dies ist in den Arbeiten von MITHÖFER (2008) und HUBER ET AL. (2008) bereits geschehen. Es werden diesen Arbeiten allerdings einige wenige Punkte entnommen und diskutiert. Meistens wird aber nur auf diese Arbeiten verwiesen. Um das Konzept der ULR in seiner Gesamtheit betrachten zu können, ist daher für den Leser der Einbezug der erwähnten Arbeiten notwendig.

Diese Arbeit enthält ausserdem keine Diskussion von der ULR zugrundeliegenden theoretischen Konzepten wie z.B. das Recht auf Grundeigentum als Rechtsbegriff und sein Einfluss auf wirtschaftliche Entwicklung oder Konzepte der Entwicklungszusammenarbeit generell. Es geht insbesondere auch nicht darum, die Diagnose und die Lösungsvorschläge von De Soto kritisch zu hinterfragen. Es ist klar, dass De Sotos Thesen in der Fachwelt nicht unwidersprochen sind (vgl. JACKSON 2002 oder ABDULAI 2006). Die Tatsache, dass sich die tansanische Regierung für ein Vorgehen gemäss De Sotos Vorschlägen entschieden hat (CLARKE ET AL: 2007:3) ist jedoch Grund genug, sich innerhalb des Frameworks von De Soto zu bewegen und die Diskussion darüber bewusst auszublenden.

Diese Arbeit richtet sich an ein wissenschaftliches Fachpublikum, das an den technischen Aspekten einer ULR interessiert ist. Gleichzeitig will diese Arbeit auch eine Hilfestellung für jene Personen sein, die sich in Zukunft mit der praktischen Umsetzung einer ULR in Tansania beschäftigen werden.

2 Grundlegende Konzepte

2.1 Universal Land Registry

Diese Arbeit entstand im Rahmen eines Projekts zum Entwurf und Aufbau einer Universal Land Registry (ULR) in Tansania. Am Anfang dieses Projekts steht ein Papier von Martin Huber (HUBER 2006a), in dem die groben Züge dieses Projekts definiert werden. Ziel dieses Projekts ist die Unterstützung des von der tansanischen Regierung eingeleiteten Formalisierungsprozesses (MKURABITA-Programm) durch den Entwurf und den Aufbau einer organisatorischen und technischen Infrastruktur für ein sicheres und transparentes Landregister. Das Projekt gliedert sich grob in drei Phasen. Während der Konzeptphase werden die rechtlichen, organisatorischen und technischen Grundlagen erarbeitet und formuliert. Worauf sie in einer zweiten Phase – der Pilotphase – in einer Testregion implementiert werden. Die dort gemachten Beobachtungen fliessen dann in die dritte Phase ein, in der die ULR institutionell und organisatorisch eingeführt und in Betrieb genommen wird.

Die vorliegende Arbeit und die parallel dazu entstehende Arbeit von Klaus Mithöfer (MITHÖFER 2008) sind in der Konzeptphase anzusiedeln. Erste Ergebnisse aus beiden Arbeiten wurden Anfang 2008 an der GSDI-10-Konferenz in Trinidad einem breiten Fachpublikum präsentiert (HUBER ET AL. 2008). Für den Bereich ULR sind somit das einführende Projektpapier, das GSDI-10-Paper, die Arbeit von Klaus Mithöfer sowie zahlreiche interne Diskussionen unter den Projektteilnehmern zentrale Grundlagen. Für eine Diskussion der ULR im Detail ist allerdings auf MITHÖFER (2008) zu verweisen. Die aus dem ULR-Konzept heraus entstehenden Anforderungen an eine technische Umsetzung werden separat in Kapitel 3 ausführlich diskutiert und festgehalten.

2.2 Modellierung von Softwaresystemen mit UML

Die Unified Modeling Language (UML) ist ein Standard der Object Management Group (OMG), mit dem objektorientierte Software modelliert werden kann (EBERLE 2006:14). UML hat sich jedoch derart weiterentwickelt, dass es in einer ganzen Reihe von Anwendungsbereichen eingesetzt wird, darunter auch in der Modellierung von Softwaresystemen und Architekturen (OMG 2005). In dieser Arbeit wird UML zur formalisierten Beschreibung der ULR-Applikation aus verschiedenen Perspektiven wie Anwender, Daten oder Architektur verwendet.

Mit UML ist es möglich, die Modellierung noch unabhängig von der späteren konkreten Umsetzung zu entwerfen und zu dokumentieren. Themen wie Programmiersprachen, Plattformen und Betriebssysteme werden bewusst ausgeklammert oder erst in einem zweiten Schritt angesprochen (SCHÄLING 2005). Es wird durch die Verwendung von UML d.h. einer formalisierten grafischen Darstellung eine gemeinsame Diskussionsbasis zwischen Entwicklern, Anwendern und Auftraggebern geschaffen.

Um ein Softwaresystem zu modellieren ist es notwendig, verschiedene Sichten auf das zu entwickelnde System festzuhalten, denn ein einfacher Anwender, der die Software bedient, hat eine ganz andere Perspektive als der Software-Entwickler, der die Software implementieren wird. Aus diesem Grund sind im UML-Standard 13 Diagrammtypen enthalten (SCHÄLING 2005). Für die Umsetzung der ULR-Applikation wurden in Anlehnung an MUTAMBO (2003:73-76) die fünf folgenden Diagramme ausgewählt:

- **Klassendiagramm**. Klassendiagramme dienen dazu, die verschiedenen Klassen, deren Aufbau und ihre Beziehungen innerhalb einer objektorientierten Software zu beschreiben. Es ist möglich, die für objektorientierte Software typischen Eigenschaften von Klassen wie Vererbung, Kapselung und Polymorphismus darzustellen (SCHÄLING 2005). Klassendiagramme werden hauptsächlich dazu verwendet, objektorientierte Software zu beschreiben. Es ist aber auch möglich, Klassendiagramme zur Beschreibung von Datenmodellen zu verwenden. Die Rolle der Klassen werden dabei von den Entitäten übernommen. Im Prinzip ist ein solches Klassendiagramm ein herkömmliches Entity-Relationship-Diagramm (ERD) in UML-Notation. Für die Beschreibung des ULR-Datenmodells wird auf ein Klassendiagramm zurückgegriffen.

- **Use-Case-Diagramm**. Mit Use Cases werden die Systemfunktionen aus Sicht der Anwender erfasst. Ein Use Case besteht aus dem Anwender (oder Akteur), dem System und dem Use Case (Funktion). Die Angaben zu den Use Cases im Diagramm werden zur besseren Lesbarkeit sehr knapp gehalten. Da dies möglicherweise dem Betrachter zu wenig Informationen bietet, wird dem Diagramm noch eine ausführlichere Beschreibung der Use Cases beigelegt. Für diese Arbeit wird in Ahlehnung an EBERLE (2006:25f) folgendes Grundschema angewendet:

Name	**Use Case-Name**
Beschreibung	Kurzbeschreibung der Funktion.
Vorbedingung	Welche Bedingungen müssen erfüllt sein, damit der Use Case ausgeführt werden kann.
Nachbedingung	Welche Bedingungen müssen nach dem erfolgreichen Durchlaufen des Use Cases erfüllt sein.
Hauptablauf	Der Ablauf des Use Cases.
alternativer Ablauf	Je nach Use Case können mehrere Varianten des Hauptablaufes vorkommen.

Fehlerablauf	Tritt beim Durchlaufen des Use Cases ein Fehler auf, wird der Fehlerablauf hier beschrieben.

Tab. 2.1: Beispielhafte Use Case-Beschreibung

Als Ergänzung zu den Use Case-Diagrammen werden Bildschirmmasken eingesetzt, mit denen das Aussehen der Applikation vorweggenommen wird. Es handelt sich dabei um graphische Darstellungen ohne Anwendungslogik. Sie dienen der Visualisierung, wie die Applikation aussehen wird. Bildschirmmasken sind kein Element des UML-Standards, werden aber häufig als sinnvolle Ergänzung zu den Use Cases eingesetzt.

- **Kompentendiagramm**. Mit dem Komponentendiagramm werden die gesammelten Anforderungen und Funktionen in Komponenten zusammengefasst. Komponenten sind Black Boxes, d.h. sie enthalten keine Informationen darüber, wie sie implementiert werden. Dafür enthalten sie Informationen darüber, welche Interfaces sie anbieten und welche Interfaces sie benötigen. (PILONE/PITMAN 2005)

- **Aktivitätsdiagramm**. Mit Aktivitätsdiagrammen werden beliebige Abläufe als Abfolge von einzelnen Aktivitäten beschrieben. Ein Ablauf beginnt immer mit einem Startpunkt und endet mit einem Endpunkt, wobei mehrere Start- und Endpunkte pro Diagramm vorkommen können. Zwischen Start- und Endpunkt werden verschiedene Aktivitäten auf bestimmte Objekte ausgeführt. Der Ablaufsteuerung dienen die Verzweigungen. Die einzelnen Elemente des Diagramms werden mit Verbindungslinien verbunden, so dass mitunter recht komplexe Abläufe in einem solchen Diagramm anschaulich dargestellt werden können. (SCHÄLING 2005)

- **Verteilungsdiagramm**. Verteilungsdiagramme zeigen die physische Ausprägung des Softwaresystems. Damit sind die konkreten Software-Teile (Programme, Bibliotheken) gemeint und wie sie auf der physisch vorhandenen Hardware verteilt sind. Es steht die Frage im Zentrum, welche Software-Komponente in welcher Beziehung mit einer anderen steht. Diese Beziehung werden mit Verbindungslinien (Kommunikationspfade) symbolisiert. (PILONE/PITMAN 2005)

2.3 Systemaufbau

Die ULR ist ein Framework, das auf verschiedenen administrativen Ebenen innerhalb von Tansania zum Einsatz kommen wird, d.h. die Spanne reicht vom lokalen Einsatzgebiet in einem Dorf bis hinauf auf die nationale Ebene. Da ist es naheliegend, dass die ULR softwareseitig nicht ein monolithischer Block sein muss, sondern aus verschiedenen miteinander kommunizierenden Komponenten auf verschiedenen Ebenen bestehen wird, wie in HUBER ET AL. (2008:12f) dargestellt. BRENTJENS (2004:7) zitiert folgende Definition für verteilte Systeme, welche auch für die ULR Gültigkeit hat:

> "A distributed system is a collection of autonomous computers linked in a network, together with software that will support integration." (WORBOYS 1995, zitiert nach BRENTJENS 2004:7)

Die technische Umsetzung muss diesen Aspekt berücksichtigen. Dabei sind auf zwei Ebenen Überlegungen anzustellen. Die eine Ebene befasst sich mit der Eigenschaft der ULR als verteiltem System. BRENTJENS (2004:7f) erwähnt drei Typen von verteilten Systemen:

- **Replizierende Systeme**. Hierbei wird der Datenbestand an mehreren Orten redundant gespeichert. Abfragen an die Daten können schnell beantwortet werden. Manipulationen an den Daten sind aber aufwendiger, da sie zuerst an jede Kopie des Datenbestandes verteilt (repliziert) werden müssen. Die Datensicherheit wird jedoch erhöht, da alle Daten mehrfach abgelegt sind.

- **Föderierte Systeme**. In föderierten Systemen werden physisch voneinander getrennte Datenbestände über gemeinsame Standards und Interfaces wie ein einziger zusammengehöriger Datenbestand behandelt und benutzt.

- **Unabhängige Systeme**. Verschiedene Datenbestände existieren unabhängig voneinander. Liegen jedoch genügend Metainformationen (z.B. über das Datenmodell und Zugriffsmöglichkeiten) vor, kann ein Anwender mehrere dieser Datenbestände gleichzeitig abfragen und nutzen.

Bei der ULR handelt es sich um ein komplexes, vielschichtiges System, das nicht einem der drei erwähnten Typen zugeordnet werden kann. Die ULR weist im Gegenteil Eigenschaften aller drei Typen auf. So sind die der Benutzerverwaltung zugrundeliegenden Daten als replizierendes System aufzubauen. Die Landregisterdaten selber werden jedoch nur auf der Ebene editiert, auf der sie auch erfasst wurden. Sie werden also in einem föderierten System vorgehalten. Zusätzlich muss die ULR auch auf Datenbestände ausserhalb der ULR zugreifen können (z.B. externe Personenregister, Geobasisdaten). Es sind damit auch Komponenten eines unabhängigen Systems anzutreffen. (HUBER ET AL. 2008:13) Es gilt daher, bei der Umsetzung eine Lösung zu finden, mit der diese drei Facetten verteilter Systeme abgedeckt werden.

Die andere Ebene fokussiert auf den Aufbau eines einzelnen Elements des verteilten Systems. Wie muss dieses gestaltet sein, damit es im verteilten System seine Rolle übernehmen kann. Der interne Aufbau ist zwar für das Funktionieren des verteilten Systems an und für sich von untergeordneter Bedeutung. Da die ULR nicht nur ein Konzept für das verteilte System ist sondern v.a. auch für das Design eines einzelnen, lokalen Knotens, muss eine technische Umsetzung auch zum internen Aufbau Überlegungen anstellen und Vorgaben machen. Der interne Aufbau eines lokalen ULR-Knotens muss folgende Kriterien erfüllen:

- **Rolle im verteilten System**. Ein lokaler Knoten muss in der Lage sein, seine Rolle in dem verteilten System zu übernehmen. Er muss mit den anderen Knoten ober- oder unterhalb in der Hierarchie auf die verabredete Art und Weise kommunizieren können. Wie oben gesehen muss er Daten replizierend öder föderierend austauschen

und auch unabhängige Datenquelle ansprechen können. Daher muss er die für die ULR geltenden Kommunikationsvorgaben d.h. Standards beherrschen und erfüllen.

- **Sicherheit**. Eine ULR kann nur erfolgreich umgesetzt werden, wenn die Bevölkerung Vertrauen in sie hat. Dies wird nur geschehen, wenn die ULR in hohem Masse sicher ist. Diese Anforderung gilt auch für die Kommunikationswege innerhalb eines lokalen ULR-Knotens. Sie müssen genau wie die Kommunikationswege des verteilten Systems gesichert sein.

- **Trennung von Daten und Funktionen**. Die zentrale Komponente der ULR sind die Daten. Diese Daten müssen auf Langfristigkeit angelegt sein. Daher macht es Sinn, die Daten von den Funktionen zu trennen. Die Daten können auf diese Weise vor wechselnden Trends im Bereich Applikation und Architektur geschützt werden.

Aus diesen Kriterien kann abgeleitet werden, dass ein lokaler Knoten modular aufgebaut sein muss und die Kommunikation zwischen den Modulen gesichert ablaufen muss. Die Kommunikation gegen aussen (d.h. zu den anderen Knoten des verteilten ULR-Systems) muss gewissen vorgegebenen Regeln folgen und ebenfalls gesichert erfolgen. Verschiedene Arbeiten (CHUNITHIPAISAN 2003:3, BRENTJENS (2004:8), RUTAMU (2006:19) und SCHÜTZE (2007:17)) haben gezeigt, dass in einem solchen Fall sehr häufig eine mehrschichtige Client-Server-Architektur gewählt wird. Eine solche Architektur teilt die Systemfunktionalitäten auf in diejenigen des Aufrufers (Client) und die des Antwortenden (Server). Sehr häufig wird zwischen diesen beiden Schichten eine Zwischenschicht (Middleware oder Applikationsschicht) eingeschaltet. Diese Zwischenschicht enthält Applikationslogik, mit der die Daten vom Server angefragt, umgewandelt und an den Client ausgeliefert werden (BRENTJENS 2004:9). Eine solchermassen erweiterte Architektur heisst 3-Schichten-Architektur. Die Auftrennung in verschiedene Schichten hat zur Folge, dass für jede Schicht verschiedene Komponenten eingesetzt und bei Bedarf auch ausgetauscht werden können. Das System ist nicht abhängig davon, welche Komponenten eingesetzt werden, sondern dass diese Komponenten die ihnen zugedachten Funktionen (s. obige Kriterien) erfüllen.

2.4 Die Rolle von Standards

Die Kommunikation zwischen den Teilen eines verteilten Systems aber auch zwischen den einzelnen Komponenten eines einzelnen Elements des verteilten Systems muss klar geregelt sein. Nun kann diese Kommunikation im Rahmen der ULR auf beliebige Weise festgelegt werden, sei es nun proprietär oder standardisiert. Sie muss nur klar spezifiziert sein. Es gibt aber zwei wichtige Anforderungen an diese Kommunikation:

- Ein Ziel dieser Arbeit ist es zu prüfen, ob die ULR mit Open Source-Komponenten umgesetzt werden kann. Dazu müssen aus dem grossen Angebot an solchen Komponenten die richtigen ausgewählt werden. Diese Komponenten, die aus verschiedenen Quellen (d.h. verschiedene Programmiersprachen, Entwicklergemeinschaften

etc.) stammen, müssen miteinander kommunizieren können. Ähnliches gilt jedoch auch für kommerzielle Produkte und Komponenten.

- Der Aufbau einer ULR in Tansania ist ein mittel- bis langfristig angelegtes Projekt. Ein Implementierungszeitraum von mehreren Jahren ist realistisch. Es wird daher notwendig sein, in Zukunft auf neue technologische Entwicklungen zu reagieren. Es ist sehr wahrscheinlich, dass der Bedarf besteht, einige der eingesetzten Komponenten gegen neuere und besser geeignete austauschen zu können, ohne dass jedoch die Kommunikation und damit das Gesamtsystem darunter leidet.

Diese beiden Anforderungen können im zentralen Begriff der Interoperabilität zusammengefasst werden. BRENTJENS (2004:11) definiert Interoperabilität folgendermassen:

> "Interoperability is the ability for a system or components of a system to provide information portability and interapplication, cooperative process control. Interoperability, in the context of the OpenGIS Specification, is software components operating reciprocally (working with each other) to overcome tedious batch conversion tasks, import/export ostacles, and distributed resource access barriers imposed by heterogenous processing environments and heterogenous data." (BUEHLER/MCKEE 1998, zitiert nach BRENTJENS 2004:11)

RUTAMU (2006:32) und BRENTJENS (2004:12) kommen zum Schluss, dass Interoperabilität nur mit Standardisierung erreicht werden kann. Denn Standards erlauben die Aufgliederung der Komplexität und zwar eine abgesprochene und verabredete Aufgliederung. Über die Standards können die beteiligten Systeme miteinander interagieren. Nur durch den Einsatz von Standards können die obenerwähnten Anforderungen erfüllt werden. Wichtig ist ausserdem, dass die Standards von Gremien oder Behörden verabschiedet werden, die von allen Beteiligten anerkannt sind und in denen sie mitwirken können. Solche Gremien gibt es sowohl im GIS-Bereich (Open Geospatial Consortium) als auch im IT-Bereich (World Wide Web Consortium, Internet Engineering Taskforce etc.). Der Einsatz von proprietären Standards einzelner Hersteller oder eigens für die ULR definierter Standards ist nicht zielführend, da damit die Auswahl an verfügbaren Komponenten stark eingeschränkt wird und in Zukunft der Austausch einzelner Komponenten nur mit sehr viel Zusatzaufwand oder gar nicht zu erreichen ist. Die oben formulierten Anforderungen können mit proprietären Standards nicht erfüllt werden. Es existieren dementsprechend auch eine Vielzahl an offenen Standards. Für die Umsetzung der ULR müssen Standards aus folgenden Anwendungsbereichen herangezogen werden:

- **sicherer Datentransport**. Jeder Datentransport zwischen den Komponenten der ULR muss gesichert ablaufen, d.h. von Unbefugten weder einseh- und manipulierbar.

- **DBMS-Zugriff**. Datenbanken machen einen wichtigen Teil der ULR aus. Daher ist der standardisierte Zugriff auf relationale DBMS unterschiedlicher Hersteller notwendig.

- **Verzeichnisdienst**. Sicherheit ist eine wichtige Anforderung der ULR. Um nachvollziehen und kontrollieren zu können, wer mit der ULR interagieren darf und wer nicht, sind Standards aus dem Bereich der Verzeichnisdienste notwendig.
- **Transaktionen**. Bei den Daten, die in der ULR anfallen, handelt es sich um komplexe Objekte, die eine räumliche Ausprägung annehmen können. Diese Daten müssen mit Transaktionen verändert werden können.
- **Kartendarstellungen**. Das Darstellen der räumlichen Daten muss ebenfalls über Standards geregelt sein.
- **Räumliche Daten in einem DBMS**. Nur die standardisierte Abfrage und Speicherung von geographischen Daten aus und in einer relationalen Datenbank gewährleistet den interoperablen Zugriff auf diese.

Die für die Umsetzung der ULR benötigten Standards werden im Folgenden ausführlicher beschrieben.

2.4.1 IT-Standards

2.4.1.1 HTTP / HTTPS

Das Hypertext Transport Protocol (HTTP) ist ein Protokoll zur Übertragung von Daten v.a. im Internet zwischen Web-Server und Web-Browser (FIELDING ET AL. 1999). Es ist ein ubiquitär eingesetzter Standard, der sich in den letzten Jahrzehnten mehr als bewährt hat. Im Rahmen einer HTTP-Verbindung werden Requests zwischen Client und Server hin und her geschickt. Diese Requests können verschiedene Ausprägungen annehmen: GET, POST, HEAD, PUT, DELETE. Damit können beliebige Daten geschickt und verlangt werden. In einer Drei-Schicht-Architektur wird HTTP v.a. für die Kommunikation zwischen Präsentations- und Applikationsschicht eingesetzt. Über das HTTP-Protokoll können auch andere, komplexere Protokolle abgewickelt werden (z.B. WFS, WMS etc.). Die Erweiterung Hypertext Transport Protocol Secure (HTTPS) ist prinzipiell identisch mit normalem HTTP, die Verbindung wird jedoch mit Secure Socket Layer (SSL) geschützt. Während dem mit HTTP die Kommunikation im Klartext zwischen Client und Server fliesst, wird sie bei HTTPS verschlüsselt. Dies wird über sogenannte Zertifikate gewährleistet. Eine Verschlüsselung erhöht natürlich die Sicherheit einer solchen Verbindung. (WIKIPEDIA 2008)

2.4.1.2 ODBC / JDBC

Die beiden Standards Open Database Connectivity (ODBC) und Java Database Connectivity (JDBC) definieren Schnittstellen, mit denen der Zugriff auf Datenbanken von beliebigen Anwendungen aus möglich wird. Damit ist es aus Sicht einer Applikation nicht mehr entscheidend, aus welchem Datenbank-Produkt sie Daten abruft. Die Applikation verwendet ausschliesslich die Funktionen der Schnittstelle und kann damit alle möglichen Datenbanken ansprechen. Der ältere Standard ODBC ist sehr weit verbreitet und fast

jedes DBMS implementiert ihn. Der neuere JDBC-Standard ist das ODBC-Pendant in der Java-Welt. (ASHENFELTER 2002)

2.4.1.3 LDAP

Das Lightweight Directory Access Protocol (LDAP) ist ein Standard, mit dem die Ressourcen eines (verteilten) Systems (Benutzer, Dienste, Drucker etc.) verwaltet werden können. Der Ort, an dem die Informationen über die Ressourcen gespeichert sind, wird Verzeichnis (Directory) genannt. Mit LDAP kann nun ein solches Verzeichnis standardisiert abgefragt und editiert werden. In dem Verzeichnis kann z.B. abgelegt werden, welcher Benutzer auf welche Teile des Systems schreibenden Zugriff hat. Dies bedeutet aber auch, dass das Verzeichnis selbst hohen Sicherheitsanforderungen genügen muss. Ein Verzeichnis, das auf LDAP basiert, kann zentral oder verteilt organisiert sein und sich wie ein oben beschriebenes verteiltes System verhalten. Im Prinzip ist mit LDAP nur das Protokoll gemeint, das den Zugriff auf das Verzeichnis beschreibt. Die Organisation des Verzeichnisses selbst wird dadurch nicht definiert. Nichtsdestotrotz wird sehr oft von LDAP-Verzeichnissen gesprochen. (JOHNER ET AL. 1998)

2.4.2 GIS-Standards

2.4.2.1 Web Map Service

Der OGC Web Map Service (WMS) ist ein relativ einfacher Standard, mit dem ein Dienst beschrieben wird, der aus geographischen Daten dynamisch (auf Anfrage) georeferenzierte Karten erzeugt. Der Dienst liefert nur das Kartenbild aus und nicht die darin enthaltenen Daten (OGC 2006a:6). Der WMS-Standard besteht aus drei Operationen: GetCapabilities, GetMap, GetFeatureInfo (optional). Erhält ein WMS-Server eine GetCapabilities-Anfrage muss er ein XML-Dokument zurückgeben, in dem er seine Metadaten (Kartenangebot, Ausgabeformate etc.) deklariert und sich damit vollständig beschreibt. Daraufhin kann ein Client mit dem Wissen aus dem GetCapabilities-Dokument einen GetMap-Request abschicken, in dem er die gewünschte Karte präzisiert. Dazu notwendige Angaben sind die Bounding Box, die sichtbaren Kartenlayer, die Grösse der Karte sowie das Format der Karte. Zumeist werden die Karten in einem der Bitmap-Formate verlangt (PNG, JPEG, GIF), es ist aber gemäss Spezifikation auch möglich, Vektorkarten im SVG-Format auszuliefern. Dies wird jedoch von den wenigsten WMS-Servern unterstützt. Danach liefert der WMS-Dienst die gewünschte Karte oder im Fehlerfall eine Fehlermeldung aus. Mit dem optionalen GetFeatureInfo-Request kann der Client Informationen zu einzelnen Features in der Karte verlangen. (OGC 2006a:21-40)

Gerade wegen seiner Einfachheit ist der WMS-Standard einer der am meisten implementierten OGC-Standards. Mit ihm können beliebige Clients mit WMS-Servern kommunizieren und Karten abrufen, unabhängig davon, wie die den Karten zugrundeliegenden Daten gespeichert sind. Dies legt einen Einsatz im Rahmen der ULR nahe.

2.4.2.2 Web Feature Service

Im Gegensatz zum WMS-Standard werden beim WFS-Standard keine grafischen Karten sondern die Daten selber übertragen, und zwar in beide Richtungen. Dies führt natürlichen zu mehr Komplexität. Der einfachere Teil des WFS-Standards befasst sich mit dem Abrufen von Features von einem WFS-Server. Ein Client kann mit WFS-Operationen Daten vom Server beziehen und weiterverarbeiten. Dies wird mit den Operation GetCapabilities (OGC 2005a:79-92) und GetFeature (OGC 2005a:32-48) abgedeckt. Die GetCapabilities-Operation hat dieselbe Bedeutung wie im WMS-Standard. Mit der GetFeature-Operation werden die gesuchten Features eingegrenzt und vom Server abgerufen. Die Eingrenzung der Features geschieht dabei über die separate Filter Encoding-Spezifikation (OGC 2005b). Hier können die Daten räumlich und über ihre Attribute eingegrenzt werden. Um eine solche Eingrenzung vornehmen zu können, benötigt der Client Informationen über die Art der verfügbaren Daten. Die holt er sich mit der DescribeFeatureType-Operation (OGC 2005a:24-31). Ein FeatureType ist eine Gruppe von Features mit gleichen Attributen und Eigenschaften. Ein WFS-Server, der nur diese Operationen unterstützt, ist ein Basic WFS-Server, von dem nur gelesen werden kann (OGC 2005a:3). Die Daten, die der WFS-Dienst ausliefert, liegen im GML-Format vor. Die Geography Markup Language (GML) ist ein XML-Dialekt, mit dem beliebig komplexe räumliche Daten modelliert und ausgetauscht werden können (OGC 2004:1).

Die WFS-Variante, die auch schreibenden Zugriff erlaubt, wird Transaction WFS-Server (WFS-T) genannt. Damit ist es möglich, Transaktionen mit dem Server abzuwickeln, d.h. die Daten, die der Server anbietet, zu verändern, zu löschen oder zu ergänzen. Dazu sind neben den bereits genannten noch weitere Operationen notwendig: GetGMLObject (optional), LockFeature (optional) und Transaction. GetGMLObject erlaubt, das Abfragen eines einzelnen Features über seine XML-ID (OGC 2005a:49-54). Mit LockFeature können einzelne Feature für andere Transaktionen gesperrt werden (OGC 2005a:55-62). Die wichtigste Operation ist jedoch die Transaction-Operation, mit der die eigentliche Transaktion abgewickelt wird. In einem Transaction-Request übermittelt der Client dem Server die Änderungen, die er an den Daten vornehmen will (OGC 2005a:63-78). Transaktionen mit einem WFS-Server sind nicht ganz unkompliziert. HUBER (2006b:5) weist drei Problembereiche aus.

- Erlaubt ein WFS-Server Transaktionen, muss er im Hintergrund seine Daten in einem transaktionsfähigen DBMS halten und nicht etwa in einer Datei, die nur einfachen Zugriff erlaubt. Dies ist mit der Unterstützung des Simple Feature-Standards (vgl. Kapitel 2.4.2.3) durch zahlreiche DBMS-Hersteller keine Hürde mehr.

- Der Zugriff auf einen Transaction WFS-Server muss über Benutzerrechte geregelt werden. Unkontrollierte Schreibzugriffe auf einen WFS-Server sind nicht wünschbar. Ein WFS-T-Server sollte immer mit einem Authentifizierungssystem gekoppelt sein.

- Daneben muss auch die Verbindungssicherheit beachtet werden. Der Server muss sicher sein, dass die Anfragen vom richtigen Urheber kommen. Dies kann mit einer verschlüsselten HTTPS-Verbindung gewährleistet werden.

Die beiden letzten Punkte sind in Zusammenhang mit dem WFS-Standard noch offene Punkte. Eine genauere Analyse ist daher notwendig. MÜHLEMANN (2007:75-81) weist fünf Ebenen aus, auf denen die Sicherheit von Geowebdiensten gewährleistet werden muss:

- **Transportsicherheit**. Mit der Transportsicherheit wird eine sichere Verbindung zwischen Client und Service angestrebt. Dies kann bei kurzfristigen Verbindungen meistens mit einer verschlüsselten HTTPS-Verbindung gewährleistet werden. Bei lang andauernden Verbindungen ist möglicherweise eine VPN-Verbingung zweckmässiger.

- **Nachrichtensicherheit**. Hier wird die Sicherheit der übermittelten Nachricht gewährleistet, indem die Nachricht selber verschlüsselt wird. Einsetzbare Standards sind XML Encryption oder XML Signature.

- **Authentifizierung**. Mit der Authentifizierung wird die Identität eines Client überprüft. Meistens geschieht dies mit Username und Passwort, es sind aber auch andere Methoden wie Smartcards oder Fingerabdrücke denkbar.

- **Autorisierung**. Die Autorisierung garantiert, dass ein Benutzer nur diejenigen Dienste und Ressourcen beanspruchen kann, für die er berechtigt ist.

- **Accounting**. Über ein Accounting wird die Nutzung der Geowebdienste protokolliert. Es wird aufgezeichnet, wer was wann gemacht hat. Damit können Transaktionen auch im Nachhinein nachvollzogen werden.

Das Zusammenspiel von Geowebdiensten und Sicherheit ist allerdings momentan noch ein offenes Feld. Es gibt verschiedene Initiativen und Standardentwürfe. So hat z.B. das OGC noch keinen Standard zu diesem Thema verabschiedet, der dieses Thema für die GIS-Welt und besonders in Zusammenhang mit WFS abschliessend behandelt. Es gibt eine Reihe von Konzepten und Standardentwürfen, die in diese Richtung gehen (WAS, WSS, GeoXACML, GeoDRM oder WPOS/XCPF). MÜHLEMANN (2007) bietet eine weiterführende Analyse der aktuellen Initiativen in diesem Bereich.

2.4.2.3 Simple Feature Access (SQL option)

Mit dem Simple Feature Access-Standard (SFS) wird festgelegt, wie in einer Datenbank abgelegte räumliche Features über die Abfragesprache SQL abgefragt und übermittelt werden. Es ist nicht Inhalt dieses Standards, die Speicherung von räumlichen Informationen in einer Datenbank festzulegen, sondern lediglich, wie diese Informationen in die Datenbank eingegeben und wieder abgefragt werden können. (HUBER 2006c:7f) Teil der Spezifikation sind Geometrietypen und SQL-Funktionen. Eine SFS-Implementation muss diverse Geometrietypen akzeptieren (z.B. Point, Linestring, Polygon, MultiPoint etc.) (OGC 2006b:36) sowie dazugehörige Funktionen wie ST_asText, ST_isSimple oder ST_Intersects (OGC 2006b:38). Räumliche Daten werden in der Datenbank als Tabellen mit einer oder mehreren Geometriespalten eines der obenerwähnten Geometrietypen definiert. Die Frage des räumlichen Bezugssystems wird so gelöst, dass einer Geometriespalte

genau ein Bezugssystem zugeordnet wird. Die Bezugssysteme werden in einer separaten Tabelle verwaltet (OGC 2006b:15). Der SFS-Standard ist somit einer der wichtigsten OGC-Standards, da er den Umgang mit den Datenquellen (Datenbanken) definiert, in denen die räumliche Information gehalten wird.

3 Anforderungen

Um eine Umsetzung der ULR zu konzipieren, ist es notwendig, alle relevanten Anforderungen zusammenzutragen und aufzulisten. Sie sind der Rahmen, in dem sich die Applikation bewegen wird. In diesem Falle handelt es sich hauptsächlich um Anforderungen, die im Rahmen der Arbeit von Klaus Mithöfer entstanden sind (MITHÖFER 2008) und den grösseren Rahmen der Universal Land Registry definieren.

3.1 Das Konzept der ULR

Das Grobkonzept der ULR, wie es HUBER ET AL. (2008:5f) definiert haben, stellt eine Reihe von grundsätzlichen Anforderungen an die tatsächliche Applikation. Daher sei dieses Grobkonzept an dieser Stelle in aller Kürze erläutert.

Der Kern der ULR ist eine geographische (räumliche) Datenbank. Diese Datenbank enthält Daten in einem allgemeinen Datenmodell, das georeferenzierte Landrechte und die dazugehörigen Informationen (Besitzer, Identitätsnachweis, Art des Rechtes) beschreibt. Die Georeferenzierung kann auf verschiedenen Genauigkeitsstufen erfolgen. Es ist durchaus möglich, dass präzise vermessene Parzellen und grobe Punktangaben nebeneinander in der Datenbank exisitieren. Jede Transaktion von Landrechten muss in der Datenbank festgehalten werden, um die Historie eines Landrechtes rückverfolgen zu können. Die ULR beinhaltet diverse Synchronisationsmechanismen, die die Daten auf die verschiedenen administrativen Ebenen (lokal, regional, national) verbreiten (verteilte Architektur). Daneben wird eine unabhängige Benutzerverwaltung mit den Funktionen Authentifizierung, Authorisation und Accounting (MÜHLEMANN 2007:48) angestrebt, in der alle berechtigten Anwender erfasst und verwaltet werden.

3.2 Die verteilte Architektur der ULR

Abbildung 3.1 zeigt die verteilte Architektur der ULR. Die ULR folgt dem Prinzip, dass alle Aktionen auf der tiefstmöglichen administrativen Ebene vorgenommen werden. Dies hat eine verteilte Architektur zur Folge. Jeder geographischen Einheit (Dorf, Quartier etc.) wird ein Knoten zugeordnet. Dabei handelt es sich um eine einfache Workstation (mit Fingerabdruckleser, Scanner, Drucker, Digitalkamera) oder auch um eine mobile Station (Laptop mit GPS-Gerät zur Datenerfassung im Feld). Eine Verbindung zu anderen ULR-Knoten muss nicht permanent gewährleistet sein, erleichtert die Arbeit jedoch. Mit diesen

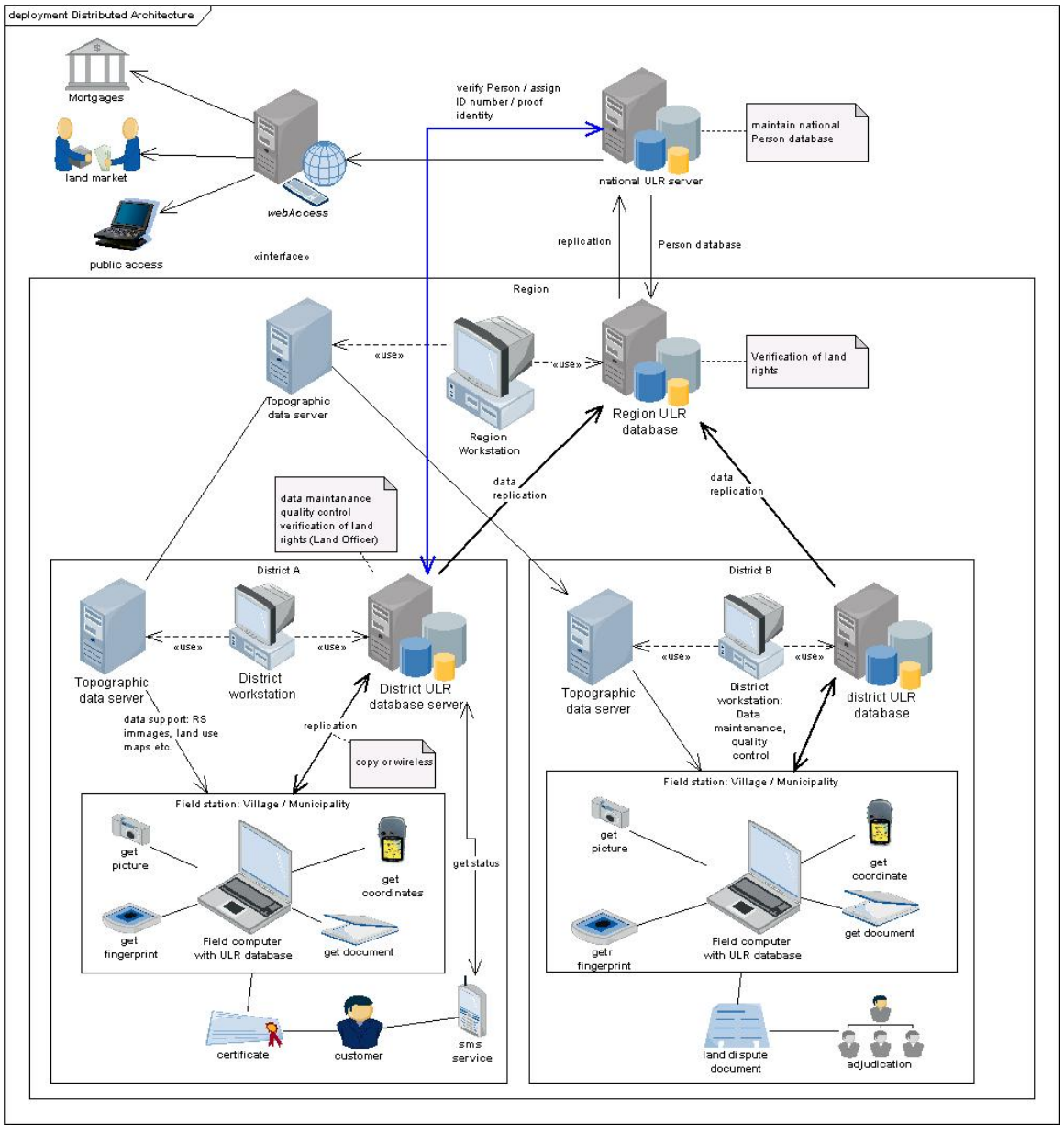


Abb. 3.1: Die verteilte Architektur der Universal Land Registry (Quelle: MITHÖFER 2008)

Basisknoten können nur diejenigen Rechte und Transaktionen registriert werden, die auch in dieser geographischen Einheit getätigt werden.

Von Zeit zu Zeit müssen die Basis-Knoten Kontakt mit einem Distrikt-Knoten aufnehmen, dabei wird der Datenbestand nach oben ausgestauscht. Dasselbe geschieht zwischen den Distrikt-Knoten und regionalen Knoten. Auf diesen Ebenen können nur in genau vorgegebenen Fällen Transaktionen vorgenommen werden. Zu diesen Fällen gehören sicherlich rechtliche Einschränkungen aufgrund von Naturschutzgebieten, überregionalen Planungsbeschlüssen etc., die in den seltensten Fällen lokal beschlossen werden, sondern immer auf regionaler wenn nicht sogar nationaler Ebene.

Zuoberst steht der nationale Knoten, der die Landrechte des gesamten Landes sammelt. Parallel zum eigentlichen Landregister gibt es ein gleichermassen verteiltes Personenregister, in der alle Anwender der ULR vom Datenerfasser, über den regionalen Manager bis zum nationalen Beamten registriert sind. Auf diesem Register baut der AAA-Dienst auf.

Die gewählte verteilte Architektur sorgt mit ihrer hohen Datenredundanz und starken Kontrollmechanismen für ein stabiles System. Ein Landrecht oder eine Transaktion wird zwar an nur einer Stelle erfasst, aber via Replikation und Synchronisation an mehreren Orten gespeichert. Das Risiko von Datenverlusten – ob bewusst oder unbewusst herbeigeführt – wird dadurch verringert.

Für die Beschreibung der ULR-Applikation ist dieser verteilte Ansatz von grosser Bedeutung. Die Applikation kann dadurch ganz auf die Bedürfnisse der Anwender auf der lokalen Ebene angepasst werden. Sie dient eben nicht der Verwaltung von Millionen von Landrechten auf nationaler Ebene, sondern der Registrierung von Rechten und Transaktionen auf lokaler Ebene. Der Anwender ist in der Regel weder IT- noch GIS-Experte. Die Applikation muss deshalb auch von Personen mit geringer IT-Ausbildung bedient werden können.

3.3 Die ULR-Prozesse

In der Arbeit von Klaus Mithöfer (MITHÖFER 2008) werden für die ULR zwei Hauptprozesse definiert.

3.3.1 Registrierung von Personen

Bevor Landrechte oder Transaktionen registriert werden können, müssen die involvierten Personen registriert werden. Abbildung 3.2 zeigt diesen Prozessablauf schematisch auf. Der Prozess wird dadurch kompliziert, dass in Tansania die wenigsten Personen über einen Personalausweis oder einen anderen staatlich ausgestellten Identitätsbeweis verfügen. Deshalb muss zu einem Umweg über eidesstattliche Erklärungen (engl: affidavits) gegriffen werden, mit denen eine Person die Identität einer anderen Person bestätigt. Zusätzlich wird die Person fotografiert und ihr werden Fingerabdrücke abgenommen. Ausserdem muss berücksichtigt werden, dass nicht nur natürliche Personen sondern auch juristische Personen (Firmen) oder gar Gruppen (Sippen, Familien, Stämme etc.) registriert werden müssen. Eine ausführliche Diskussion des Prozesses liefert MITHÖFER (2008).

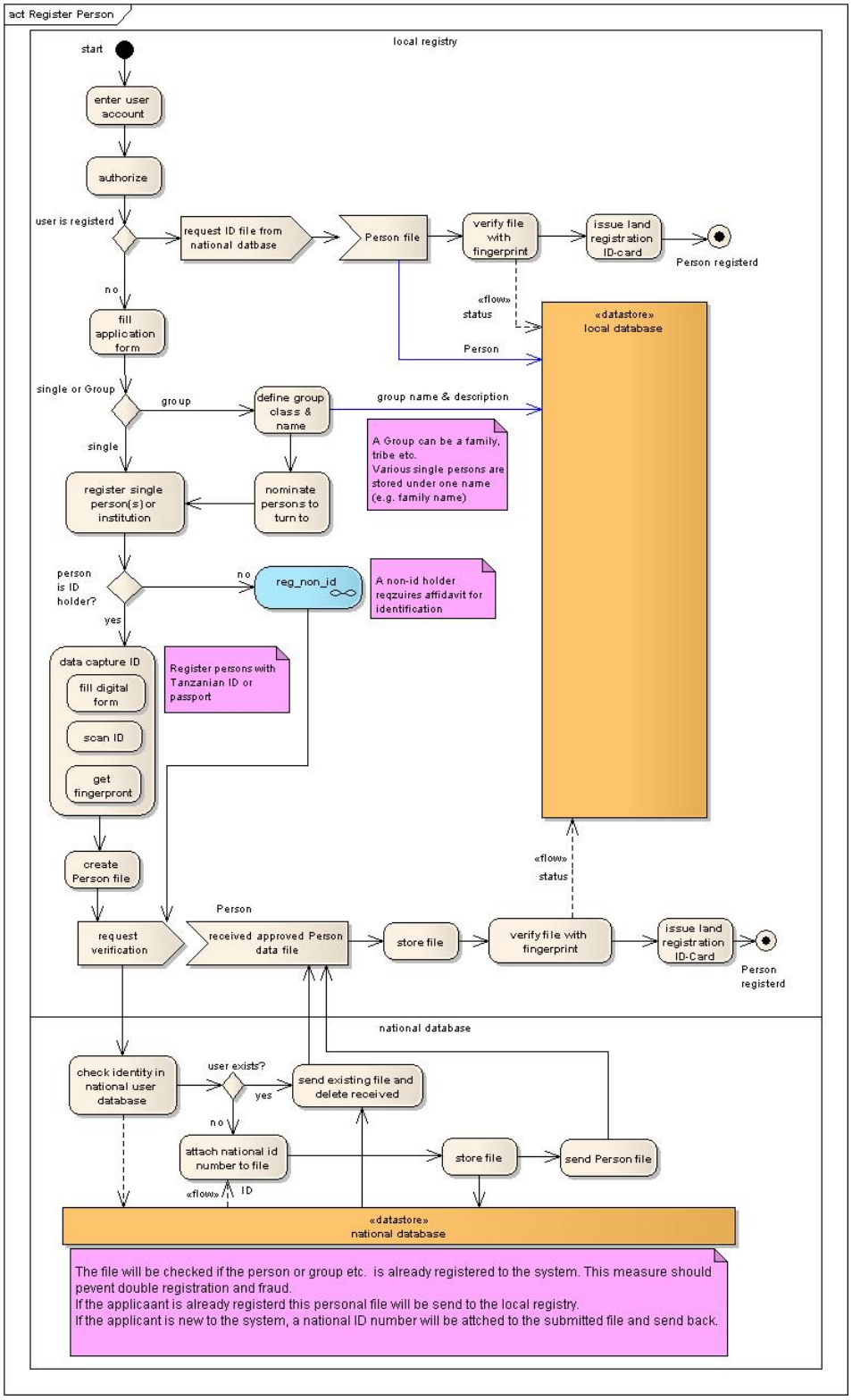


Abb. 3.2: Der Prozess Register Person (Quelle: MITHÖFER 2008)

3.3.2 Registrierung von Landansprüchen

Wenn alle notwendigen Personen einwandfrei erfasst sind, kann die Parzelle und das dazugehörige Landrecht erfasst werden. Aus Abbildung 3.3 wird die notwendige Vorgehensweise ersichtlich. Die wichtigsten Schritte sind die Georeferenzierung der Parzelle, die Zuordnung des Landrechts zu einer registrierten Person und das Aufzeigen von Landkonflikten. Dabei sind je nach Landrechtstyp verschiedene Angaben zu machen. Auch bereits bestehende Dokumente (Besitzurkunden oder informelle Dokumente) werden registriert und archiviert. Im Falle eines Landkonfliktes wird bereits bei der Eingabe darauf hingewiesen und anschliessend eine kurze Dokumentation ausgedruckt, um damit eine erste Grundlage für eine Beilegung des Konflikts zu legen. Um die Nachvollziehbarkeit zu gewährleisten, werden an bestimmten Zeitpunkten im Prozess Dokumente ausgedruckt, die als Beleg über die getätigte Registrierung dienen.

Mit diesem Prozess können verschiedenste Arten von Dokumenten und Ansprüchen registriert werden, dazu gehören insbesondere auch die Transaktionen, d.h. den Verkauf eines Grundstücks an eine andere Person. Die Transaktionen müssen daher nicht gesondert beschrieben und modelliert werden. Der in Abbildung 3.3 gezeigte Prozesse enthält mehrere Subprozesse, die an dieser Stelle nicht näher erläutert werden. Sie werden bei MITHÖFER (2008) diskutiert.

3.3.3 Konsequenzen für die Umsetzung

Eine zentrale Anforderung an die ULR-Applikation ist es nun, den Anwender beim Durchlaufen dieser Prozesse so gut wie möglich zu unterstützen. Vom Anwender kann nicht verlangt werden, dass er alle Prozesse permanent in Erinnerung hat. Die Anwendung muss ihm vielmehr immer nur die gerade notwendigen Informationen anzeigen und darauf achten, dass er alle notwendigen Informationen eingibt. Damit kann verhindert werden, dass Landrechte unvollständig registriert werden. Gleichzeitig wird garantiert, dass die Prozesse von unterschiedlichen Personen immer gleich durchlaufen werden.

Eine genaue Analyse der Prozesse zeigt, dass die Applikation Daten in verschiedenster Art vom Anwender akzeptieren muss. Die Palette reicht von der einfachen Textinformation über Datumsinformationen bis hin zu Fotografien oder Scans (Fingerabdrücke, Dokumente). Für die Ausgabe sind verschiedene Berichte zu definieren, die der Anwender per Knopfdruck ausdrucken und dem Landbesitzer abgeben kann.

3.4 Das ULR-Datenmodell

Eine ausführliche Beschreibung des ULR-Datenmodells findet sich in MITHÖFER (2008). An dieser Stelle soll auf die wichtigsten Punkte hingewiesen werden. Das Datenmodell stützt sich in weiten Teilen auf die schon weit fortgeschrittenen Arbeiten zur Erarbeitung eines allgemeinen Modells zur Erfassung von Grundeigentumsinformationen. Dieses allgemeine Modell ist das Core Cadastral Domain Model (CCDM) und wurde in einer FIG-Arbeitsgruppe entwickelt. Das CCDM liegt mittlerweile in der Version 1.0 vor (LEM-

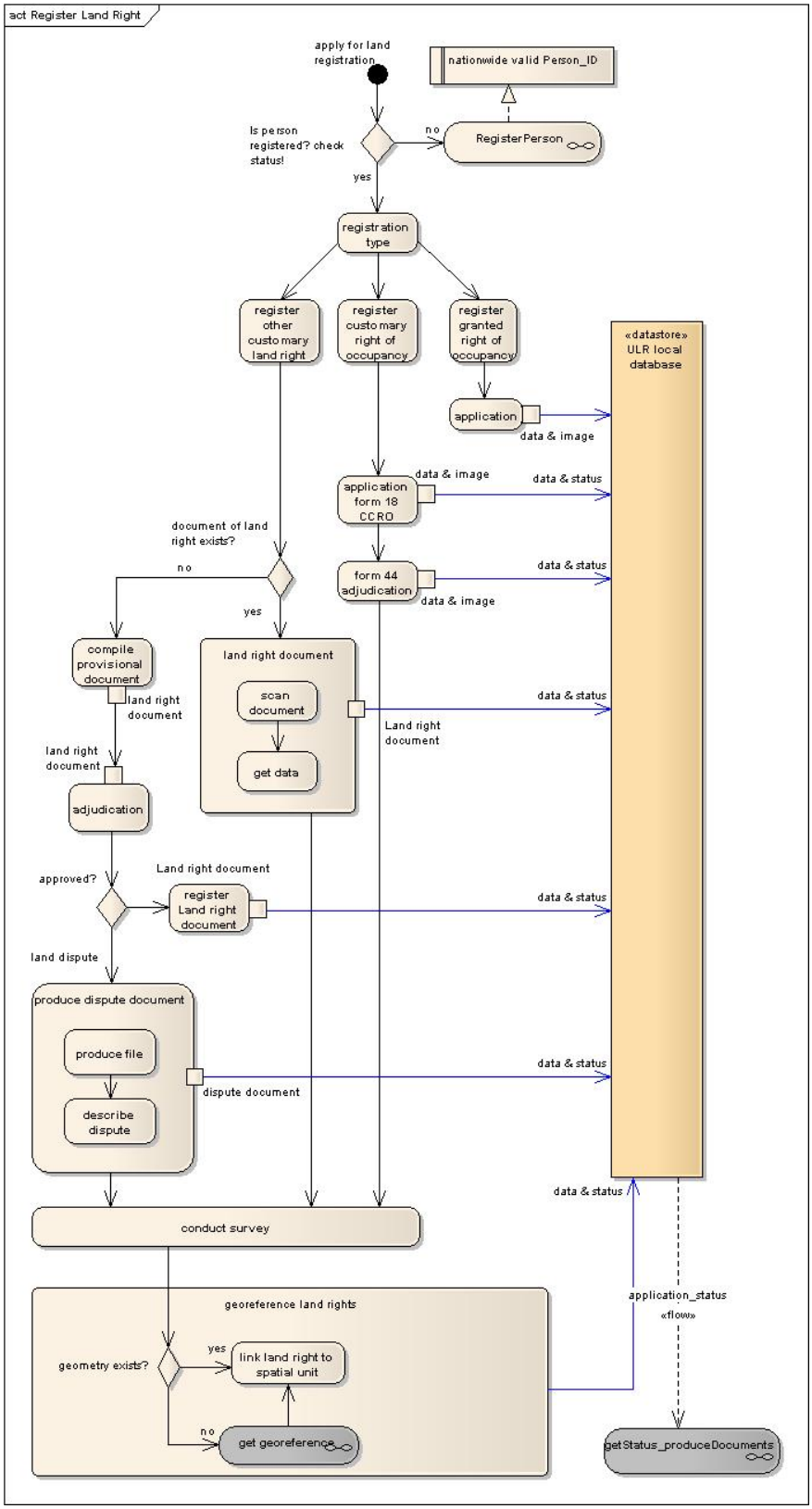


Abb. 3.3: Der Prozess Register Land Right (Quelle: MITHÖFER 2008)

MEN/VAN OOSTEROM 2006). Im Zentrum des CCDM stehen die drei Klassen RegisterObject (Parzelle, Grundstück), Person (natural, nonnatural, group) und RRR (Right, Restriction, Responsibility). Dazu kommt eine Vielzahl an weiteren Klassen, mit denen beliebig komplexe Fälle abgedeckt werden können. Das CCDM selber basiert wiederum auf den Vorgaben von CADASTRE 2014 (KAUFMANN/STEUDLER 1998). Als Weiterentwicklung des CCDM wird zurzeit das Social Tenure Domain Model (STDM) definiert. Es ist eine Spezialisierung des CCDM für Entwicklungsländer (LEMMEN ET AL. 2007:1).

Das Datenmodell der ULR übernimmt nun die grundlegenden Ansätze und Klassen des CCDM bzw. des STDM, indem es die drei Kernklassen RegisterObject, Person und RRR aufnimmt. Abbildung 3.4 zeigt dieses ULR-Datenmodell in vereinfachter Form. Das Modell beschreibt die Beziehung zwischen Parzelle und Person über eine Land Tenure-Klasse, mit der alle Rechte, Verantwortlichkeiten und Einschränkungen (RRR) modelliert werden. Diese Klasse enthält alle Informationen u.a. in Form von gescannten Besitzurkunden (informeller oder formeller Art) über die Art des Landbesitzes (Eigentum, Pacht, Nutzungsrecht etc.). Die betroffene Parzelle wird mit der Klasse Spatial Unit beschrieben. Abbildung 3.5 zeigt deutlich, dass eine Spatial Unit keineswegs nur ein topologisch korrektes Polygon sondern auch eine bewusst unpräzise Linien- oder Punktinformation sein kann. Ebenfalls in dieser Klasse enthalten sind weitere Beschreibungen des betroffenen Landstücks (Kartenskizzen, Textbeschreibungen, Fotografien etc.). Damit ist eine umfassende Identifikation eines Landstücks möglich. Die Klasse Person beinhaltet Informationen über den Besitzer eines Landrechts aber auch über den Aussteller einer eidesstattlichen Erklärung.

3.4.1 Konsequenzen für die Umsetzung

Die ULR-Applikation muss nun den Anwender in die Lage versetzen, die Vorgaben des Datenmodells erfüllen zu können. Sie muss ihn dabei unterstützen, die richtigen Eingaben zu machen und keine Attribute zu vergessen. Das Datenmodell muss in der ULR-Applikation umgesetzt werden.

3.5 Standardunterstützung

Die ULR ist ein Projekt zur Erfassung einer Informationsmenge, die auch für eine ganze Reihe weiterer Politikbereiche wie z.B. Raumplanung, Steuerwesen, Umweltschutz etc. interessant ist. Deshalb muss gewährleistet sein, dass diese Bereiche ebenfalls – unter kontrollierten Bedingungen (Zugriffsrechte) – auf die ULR-Daten zugreifen können. Nur so können die ULR-Daten umfassend genutzt werden. Um dies zu erreichen, muss der Zugang zu den ULR-Daten über standardisierte Protokolle und Formate ermöglicht werden. Da es momentan in Tansania noch keine funktionierende nationale Geodateninfrastruktur (NGDI) gibt sondern nur erste Aufbauaktivitäten (vgl. JOHANSSON 2005), ist es zweckmässig, wenn sich die ULR-Applikation auf die etablierten OGC-Standards stützt. Somit ist es eine wichtige Anforderung, dass auf die Daten der ULR über Standards wie WFS und/oder WMS zugegriffen werden kann.

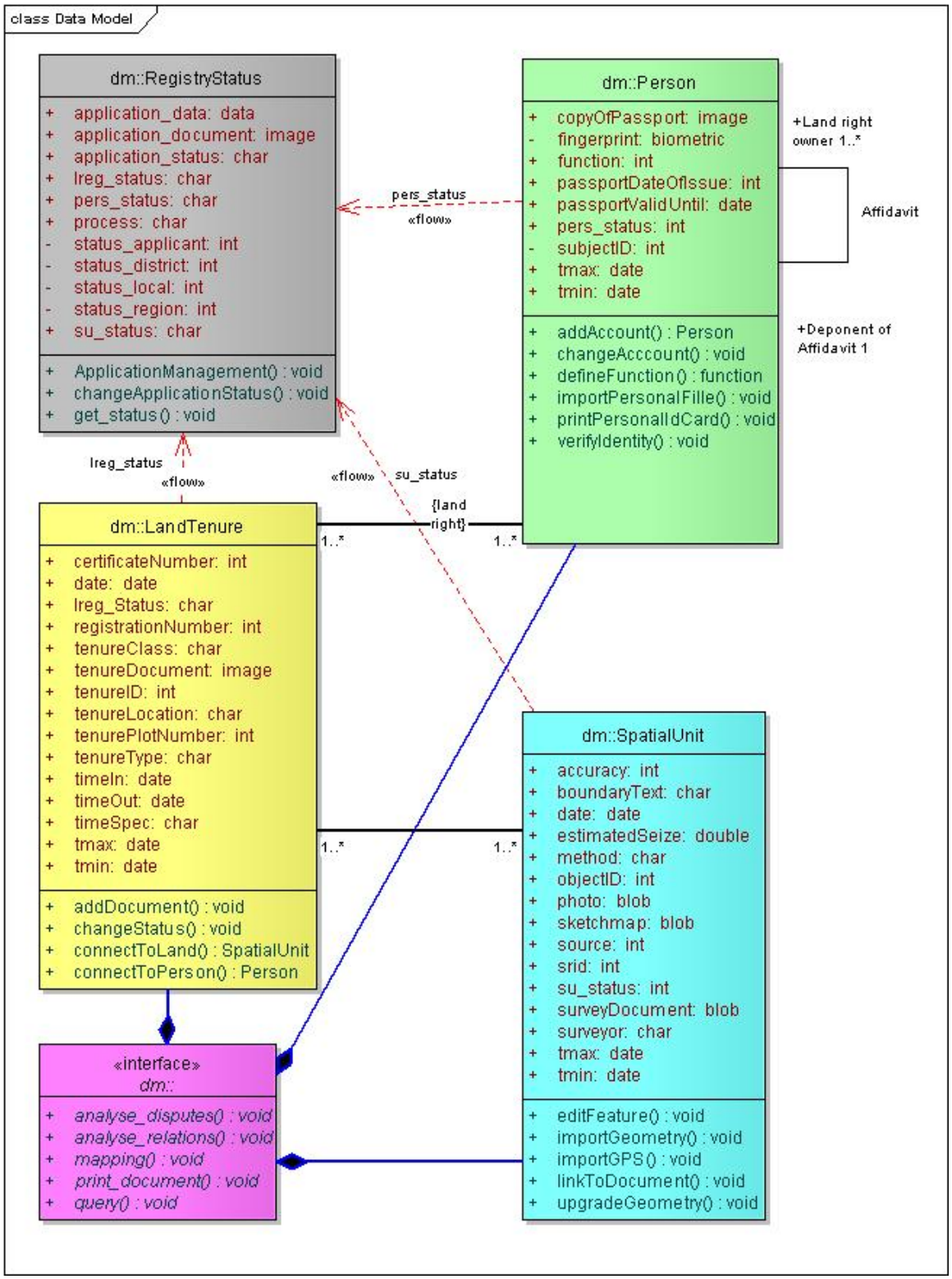


Abb. 3.4: Das ULR-Datenmodell in vereinfachter Form (Quelle: MITHÖFER 2008)

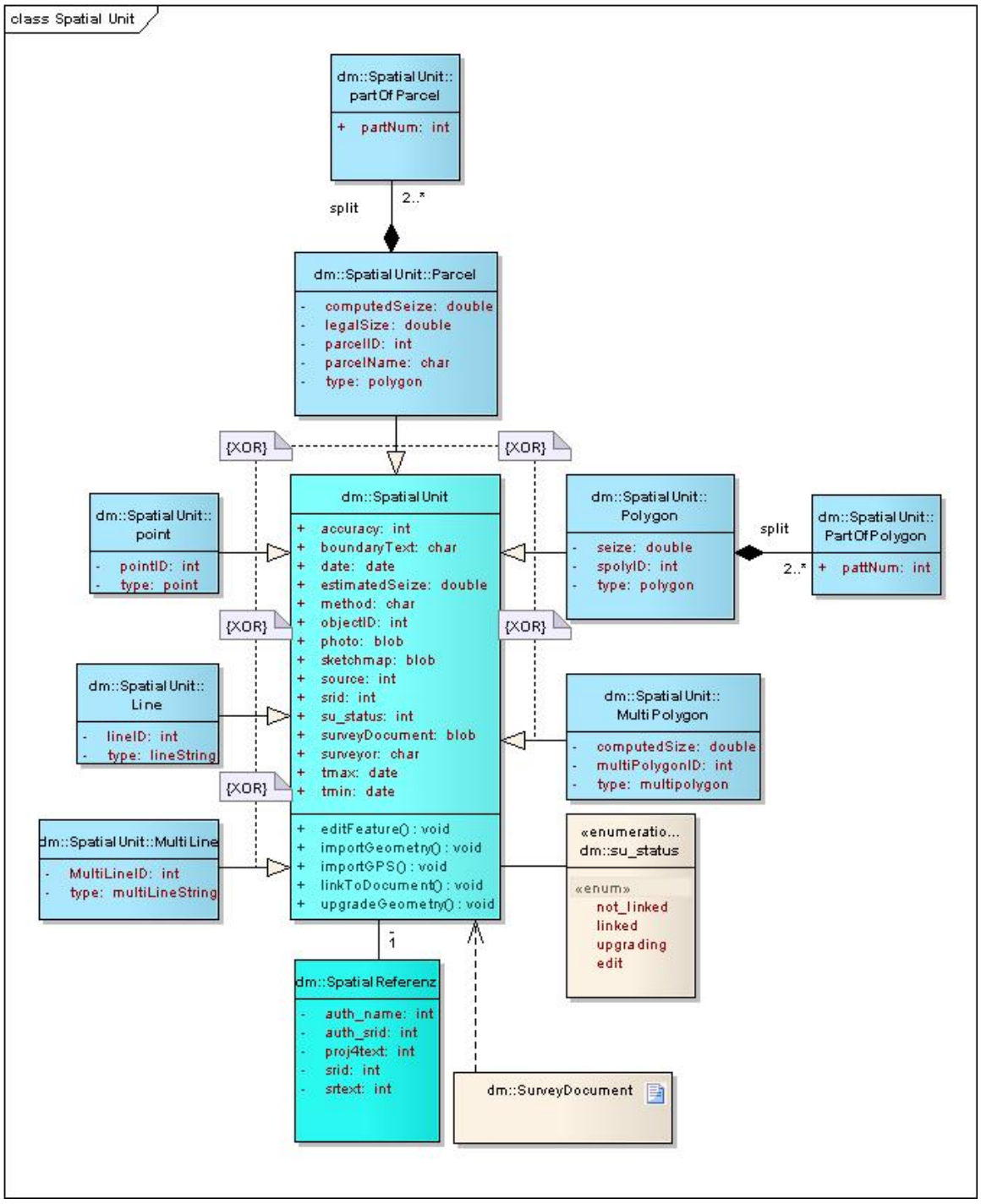


Abb. 3.5: Die Klasse Spatial Unit (Quelle: MITHÖFER 2008)

Wie die Kapitel 2.3 und 2.4 gezeigt haben, ergibt sich aber v.a. aufgrund der verteilten Architektur der ULR, dass zwingend Standards für die Kommunikation zwischen den Systemkomponenten eingesetzt werden müssen.

3.6 Einfachheit

Wie in Abschnitt 3.3 bereits erwähnt, muss davon ausgegangen werden, dass viele Anwender der ULR-Applikation weder GIS- noch IT-Spezialisten sind und nur über eine rudimentäre Ausbildung in diesen Bereichen verfügen. Daraus ergibt sich die Anforderung, dass die ULR-Applikation möglichst einfach aufgebaut sein muss, damit auch Personen mit wenig IT- oder GIS-Erfahrung in der Lage sind, sie zu bedienen. Die Forderung nach Einfachheit ist auch in der Analyse von De Soto wiederzufinden (DE SOTO 2006:56).

3.7 Umsetzung mit Open Source Software

In Anbetracht der Tatsache, dass mittlerweile eine ganze Reihe von Open Source-Produkten im GIS-Bereich existieren und diese so gut wie alle denkbaren Anforderungen im GIS-Bereich erfüllen können (vgl. RAMSEY 2007), macht es sicherlich Sinn, den Einsatz von Open Source-Produkten für die Implementierung einer ULR-Applikation als Anforderung festzuhalten. Dies hat nicht zuletzt auch finanzielle Aspekte, denn für DE SOTO (2006:56) ist eine "low cost"-Lösung erforderlich. Die Forderung nach "low cost" bezieht sich bei De Soto zwar auf den Preis, den ein Bürger für eine Registration zu bezahlen hat, sie kann aber auch so verstanden werden, dass das zu implementierende System selbst günstig im Aufbau und Betrieb zu sein hat. Ausserdem hat Open Source den Vorteil, dass der Quellcode offen zugänglich ist. Dies trägt zu einer höheren Transparenz der ULR bei.

3.8 Sicherheit

Wie in Abschnitt 2.4.2.2 gezeigt, gibt es beim Einsatz von Geowebdiensten eine Reihe von Anforderungen im Sicherheitsbereich. Die fünf Sicherheitsebenen (Transportsicherheit, Nachrichtensicherheit, Authentifizierung, Autorisierung und Accounting) müssen in der ULR umgesetzt werden. Die Akzeptanz der ULR steht und fällt mit ihrer Sicherheit und Transparenz. Deshalb muss bei der Umsetzung sehr darauf geachtet werden. Für die ULR gilt allgemein die Anforderung, dass das gesamte System sicher sein muss (HUBER 2006a:1).

3.9 Zusammenfassung

Kurz zusammengefasst existieren folgende zentrale Anforderungen an die Umsetzung der ULR:

- ULR-Konzept (verteilte Architektur)

- ULR-Prozesse
- ULR-Datenmodell
- Unterstützung von IT- und GIS-Standards
- einfache Benutzerführung
- Einsatz von Open Source-Software
- Sicherheit

4 Use Cases

Da es sich bei der ULR um ein neu zu entwickelndes System handelt und nicht um eine bestehende Applikation, ist es nur sehr schwer möglich, tatsächliche Anwender nach ihren Anforderungen und Wünschen an die Applikation zu befragen. Auch die grosse räumliche Distanz zu Tansania macht eine Befragung potenzieller Anwender unmöglich. Deshalb sind die hier formulierten Use Cases Annahmen und keine Erfahrungswerte. Aus den in Kapitel 3 erfassten Anforderungen ergeben sich jedoch eine Vielzahl an Funktionen, die ein Anwender an die ULR haben wird, so dass die Anwendung von Use Case auch ohne Anwenderbefragung Sinn macht.

Zunächst werden die verschiedenen Akteure identifiziert und beschrieben. Die Use Cases selber sind in logisch zusammengehörigen Modulen gruppiert worden. Diese Gruppierung findet sich auch in der logischen Systemarchitektur wieder (Kapitel 5). Pro Modul wurde ein Use Case-Diagramm entworfen. Dieses ist immer der Ausgangspunkt der Beschreibungen. Nach dem Diagramm werden die einzelnen Use Cases in einer strukturierten tabellarischen Form ausführlich beschrieben.

4.1 Akteure

Es wurden vier Akteure identifiziert, die mit der ULR-Applikation interagieren.

4.1.1 Registrar

Der Registrar ist der wichtigste Akteur in der gesamten Applikation. Er gibt zusammen mit dem Landbesitzer die Daten zum Landrecht in die ULR-Datenbank ein. Er kommuniziert mit dem Landbesitzer, ermittelt die notwendigen Angaben und Dokumente zur Identifikation der Parzelle, des Rechtes und des Besitzers und speist diese in die ULR ein. Ausserdem druckt er die verschiedenen Bestätigungsdokumente aus und händigt diese dem Besitzer und anderen Betroffenen aus. Der Registrar erhält somit lesenden und schreibenden Zugriff auf die Datenbank.

4.1.2 Viewing User

Ein Viewing User ist ein beliebiger Anwender, der lesenden Zugriff auf die Datenbank hat. Die ULR-Applikation bietet Möglichkeiten auf, wie nach Parzellen, Personen und Rechten gesucht werden kann. Diese Möglichkeiten stehen dem Viewing User zur Verfügung. Nicht

inbegriffen in der Akteurgruppe Viewing User sind alle Zugriffe, die z.B. via WMS auf die ULR-Daten zugreifen. Die Vielfalt dieser Anwender ist zu hoch, um in diesem Dokument detailiert beschrieben zu werden.

4.1.3 System Administrator

Der System Administrator kümmert sich um die Stellung der einzelnen ULR-Applikation in der gesamten verteilten Architektur (vgl. Abschnitt 3.2). Konkret heisst das, er steuert die Synchronisation und Replikation des lokalen ULR-Knotens mit den Knoten oberhalb oder unterhalb in der Hierarchie. Er löst z.B. eine Replikation mit dem Knoten oberhalb aus, wenn der lokalen Knoten wieder Verbindung zu den anderen Knoten herstellen kann.

4.1.4 Data Administrator

Der Data Administrator kümmert sich um die Einbindung von Daten aus, die für das Gebiet, in dem sich der lokale ULR-Knoten sich befindet, nützlich sind. Hierbei handelt es sich z.B. um Luftbilder, Points of Interest, die zusammen mit der lokalen Bevölkerung digitalisiert werden etc. Er bindet diese Daten ein und stellt sie den Anwendern in der ULR-Applikation zur Verfügung.

4.2 Daten erfassen

Diese Use Case-Gruppe beinhaltet alle Formen von Datenerfassung, d.h. die Registrierung von Personen, die Registrierung von Grundeigentum oder Nutzungsrechten sowie die Erfassung von Basisdaten (Luftbilder, POI etc.).

4.2.1 Personen registrieren

Diese Use Cases setzen den in Abschnitt 3.3 erwähnten Prozess der Erfassung von Personen um. Dabei handelt es sich nicht nur um natürliche Personen sondern auch um juristische Personen (Firmen, Vereine) oder Personengruppen (Sippen, Stämme, Dorfgemeinschaften). Deshalb sind hier auch verschiedene Vorgehensweisen vorgesehen. Abbildung 4.1 zeigt alle Use Cases in Zusammenhang mit der Registrierung von Personen. Nachfolgend werden die einzelnen Use Cases detailliert beschrieben:

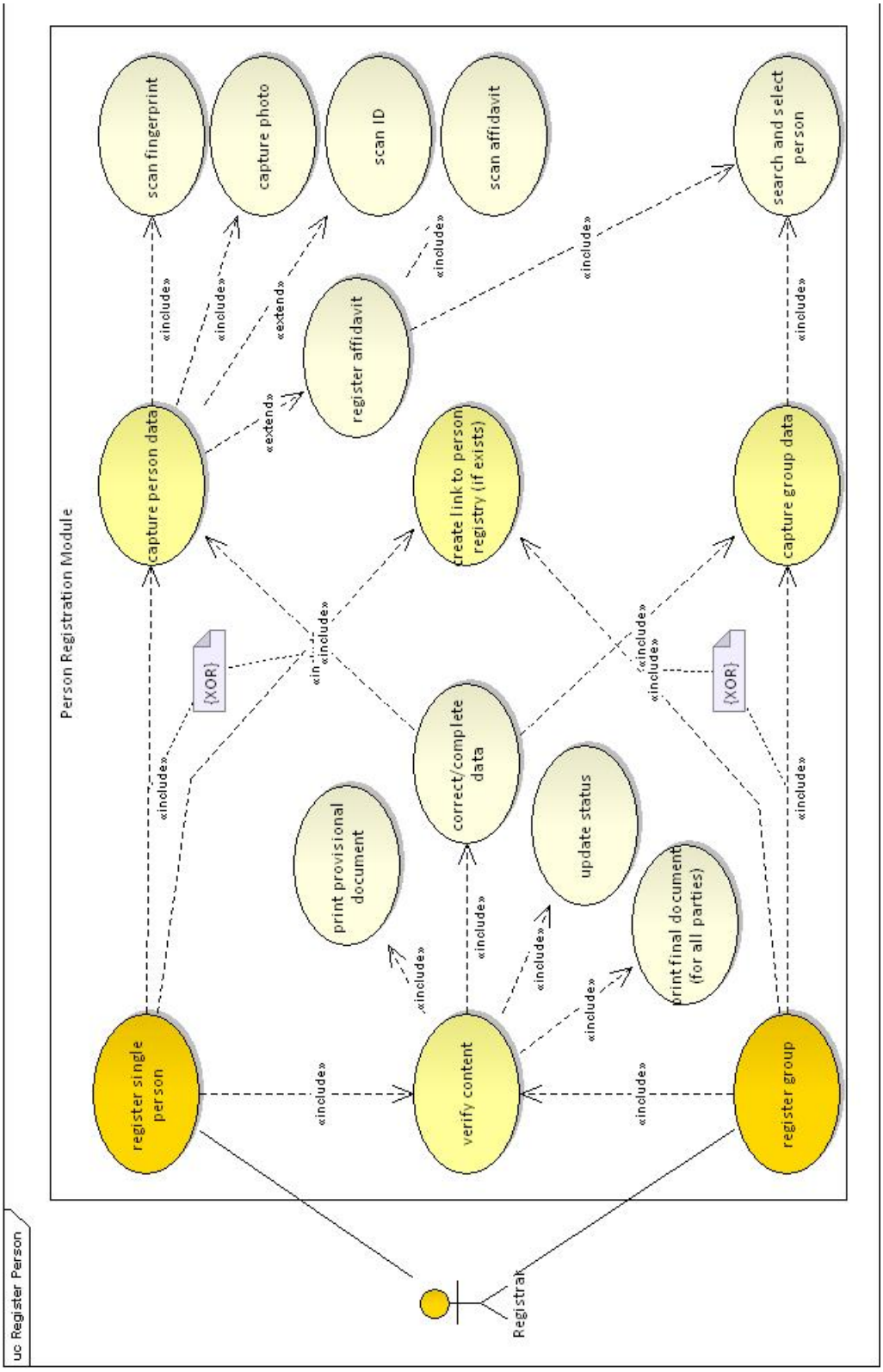


Abb. 4.1: Die Use Cases zur Registrierung von Personen

Name	**register single person**
Beschreibung	Der Registrar registriert eine einzelne Person in der ULR. Dies kann eine natürliche oder eine juristische Person sein.
Vorbedingung	Der Registrar ist korrekt am System angemeldet und hat die erforderlichen Rechte.
Nachbedingung	Die einzelne Person ist vollständig in der ULR registriert. Dies bedeutet einerseits, dass alle notwendigen Angaben zu der Person (gemäss Datenmodell) eingegeben wurden und andererseits, dass die Identität der Person erfolgreich überprüft wurde.
Hauptablauf	Der Registrar erfasst zuerst die notwendigen Angaben zur Person (Use Case *capture person data* und prüft anschliessend mit dem Use Case *verify content* die Identität der Person.
alternativer Ablauf	Ist in einem existierenden Personenregister (z.B. staatliches Personenregister) bereits ein Eintrag zu der Person vorhanden, wird mit dem Use Case *create link to person registry* nur ein Link zu diesem Eintrag in die ULR übernommen. Anschliessend wird mit dem Use Case *verify content* eine Bestätigung zu Handen der registrierten Person ausgestellt. Auf eine Prüfung der registrierten Angaben kann hier verzichtet werden.
Fehlerablauf	keiner

Tab. 4.1: Use Case *register single person*

Name	**create link to person registry**
Beschreibung	Es wird ein Link zu einem externen Register erzeugt.
Vorbedingung	Es existiert ein externes Register (z.B. ein staatliches Personenregister, Handelsregister), das für die Zwecke der ULR zugänglich ist.
Nachbedingung	Der eindeutige Schlüssel der Person aus dem externen Register ist in der ULR eingetragen.
Hauptablauf	Der Registrar ermittelt in dem externen Register den eindeutigen Schlüssel der zu registrierenden Person. Das Vorgehen hierzu ist abhängig von den Zugriffsmöglichkeiten auf dieses Register. Anschliessend trägt der Registrar den eindeutigen Schlüssel sowie eine Angabe über das verwendete externe Register in die ULR ein.
alternativer Ablauf	je nach Zugangsmöglichkeiten zu dem externen Register
Fehlerablauf	Wird die Person in dem externen Register nicht gefunden, wird dieser Use Case abgebrochen.

Tab. 4.2: Use Case *create link to person registry*

Name	**capture person data**
Beschreibung	Alle notwendigen Angaben zu der zu registrierenden Person werden in die ULR eingegeben.
Vorbedingung	Die zu registrierende Person ist in keinem zugänglichen externen Register bereits registriert.
Nachbedingung	Alle notwendigen Angaben gemäss Datenmodell zu der Person sind in der ULR registriert. Der Registrierungsstatus der Person ist auf "provisorisch" gesetzt.
Hauptablauf	Der Registrar gibt die notwendigen Angaben gemäss Datenmodell (z.B. Name, Vorname, Adresse, Telefonnummer etc.) in das von der Applikation angegebene Formular ein. Sind alle Eingaben korrekt und vollständig, werden mit den Use Cases *scan fingerprint*, *capture photo*, *scan ID* und *register affidavit* weitere Angaben zur Bestätigung der Identität registriert.
alternativer Ablauf	keiner
Fehlerablauf	Gibt der Registrar unvollständige oder falsche Angaben in das Formular ein, wird das Formular von der Applikation zusammen mit einer passenden Fehlermeldung erneut ausgegeben. Der Registrar muss die gemachten Angaben vervollständigen oder korrigieren.

Tab. 4.3: Use Case *capture person data*

Name	**scan fingerprint**
Beschreibung	Der zu registrierenden Person wird ein Fingerabdruck abgenommen und registriert.
Vorbedingung	An der Workstation, in der die ULR-Applikation verwendet wird, ist ein Fingerabdruckscanner angeschlossen.
Nachbedingung	Der Fingerabdruck ist in der ULR gespeichert.
Hauptablauf	Der Registrar digitalisiert den Fingerabdruck der Person mit einem Fingerabdruckscanner, so dass der Abdruck als Bilddatei vorliegt. Anschliessend wird die Bilddatei in das von der Applikation ausgegebene Formular eingetragen und in die ULR gespeichert.
alternativer Ablauf	keiner
Fehlerablauf	Ist die Bilddatei zu gross oder liegt sie nicht im korrekten Format vor, wird dem Registrar das Formular zusammen mit einer entsprechende Fehlermeldung erneut angezeigt.

Tab. 4.4: Use Case *scan fingerprint*

Name	**capture photo**
Beschreibung	Eine digitale Fotografie der zu registrierenden Person wird gemacht und registriert.
Vorbedingung	An der Workstation, in der die ULR-Applikation verwendet wird, ist eine digitale Kamera angeschlossen.
Nachbedingung	Das Bild ist in der ULR gespeichert.
Hauptablauf	Der Registrar fotografiert die Person und lädt die daraus entstandene Bilddatei auf die Workstation. Anschliessend wird die Bilddatei in das von der Applikation ausgegebene Formular eingetragen und in die ULR gespeichert.
alternativer Ablauf	keiner
Fehlerablauf	Ist die Bilddatei zu gross oder liegt sie nicht im korrekten Format vor, wird dem Registrar das Formular zusammen mit einer entsprechende Fehlermeldung erneut angezeigt.

Tab. 4.5: Use Case *capture photo*

Name	**scan ID**
Beschreibung	Ein Scan der ID der zu registrierenden Person wird gemacht und registriert.
Vorbedingung	Die Person verfügt über eine amtliche ID (Pass oder Personalausweis). An der Workstation, in der die ULR-Applikation verwendet wird, ist ein Dokumentenscanner angeschlossen.
Nachbedingung	Der Scan der ID ist in der ULR gespeichert
Hauptablauf	Der Registrar scannt die ID mit dem Dokumentenscanner ein, so dass der Scan als Bilddatei vorliegt. Anschliessend wird die Bilddatei in das von der Applikation ausgegebene Formular eingetragen und in die ULR gespeichert.
alternativer Ablauf	keiner
Fehlerablauf	Ist die Bilddatei zu gross oder liegt sie nicht im korrekten Format vor, wird dem Registrar das Formular zusammen mit einer entsprechende Fehlermeldung erneut angezeigt.

Tab. 4.6: Use Case *scan ID*

Name	**register affidavit**
Beschreibung	Eine eidesstattliche Erklärung über die Identität der zu registrierenden Person wird gescannt und registriert.
Vorbedingung	Eine andere Person ist bereit, die Identität der zu registrierenden Person schriftlich mittels eidesstattlicher Erklärung zu bestätigen. Diese Person ist in der ULR registriert. Die Erklärung liegt schriftlich vor.
Nachbedingung	Die eidesstattliche Erklärung ist in der ULR gespeichert.
Hauptablauf	Der Registrar sucht die Person, die die Erklärung abgibt, in der ULR (Use Case *search and select person*). Anschliessend wird die Erklärung gescannt und in der ULR gespeichert (Use Case *scan affidavit*).
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.7: Use Case *register affidavit*

Name	**scan affidavit**
Beschreibung	Ein Scan der eidesstattlichen Erklärung wird gemacht und registriert.
Vorbedingung	Die eidesstattlicher Erklärung liegt schriftlich vor und an der Workstation, in der die ULR-Applikation verwendet wird, ist ein Dokumentenscanner angeschlossen.
Nachbedingung	Der Scan der eidesstattlichen Erklärung ist in der ULR gespeichert.
Hauptablauf	Der Registrar scannt die Erklärung mit dem Dokumentenscanner ein, so dass der Scan als Bilddatei vorliegt. Anschliessend wird die Bilddatei in das von der Applikation ausgegebene Formular eingetragen und in die ULR gespeichert.
alternativer Ablauf	keiner
Fehlerablauf	Ist die Bilddatei zu gross oder liegt sie nicht im korrekten Format vor, wird dem Registrar das Formular zusammen mit einer entsprechende Fehlermeldung erneut angezeigt.

Tab. 4.8: Use Case *scan affidavit*

Name	**register group**
Beschreibung	Eine Gruppe von Personen wird in der ULR registriert. Dabei kann es sich um unterschiedliche Gruppen wie Sippen, Familien, Clans oder Dorfgemeinschaften handeln.

Vorbedingung	Der Registrar ist korrekt am System angemeldet und hat die erforderlichen Rechte.
Nachbedingung	Die Gruppe ist mit allen notwendigen Angaben zur Identität (gemäss Datenmodell) in der ULR registriert.
Hauptablauf	Der Registrar registriert die Gruppe mit dem Use Case *capture group data*
alternativer Ablauf	Ist in einem existierenden externen Register bereits ein Eintrag zu der Gruppe vorhanden, wird mit dem Use Case *create link to person registry* nur ein Link zu diesem Eintrag in die ULR übernommen. Anschliessend wird mit dem Use Case *verify content* ein Bestätigung zu Handen der registrierten Gruppe ausgestellt. Auf eine Prüfung der registrierten Angaben kann hier verzichtet werden.
Fehlerablauf	keiner

Tab. 4.9: Use Case *register group*

Name	**capture group data**
Beschreibung	Alle notwendigen Angaben zu der zu registrierenden Gruppe werden in der ULR registriert.
Vorbedingung	Die Gruppe hat eine Ansprechsperson nominiert, die bereits in der ULR registriert ist.
Nachbedingung	Die Gruppe ist mit allen notwendigen Angaben in der ULR registriert. Der Registrierungsstatus der Gruppe ist auf "provisorisch" gesetzt.
Hauptablauf	Zunächst gibt der Registrar alle Angaben zu der Gruppe (gemäss Datenmodell) in das von der Applikation ausgegebene Formular ein. Anschliessend wählt er mit dem Use Case *search and select person* die Ansprechperson der Gruppe aus den bereits registrierten Personen aus.
alternativer Ablauf	keiner
Fehlerablauf	Gibt der Registrar unvollständige oder falsche Angaben in das Formular ein, wird das Formular von der Applikation zusammen mit einer passenden Fehlermeldung erneut ausgegeben. Der Registrar muss die gemachten Angaben vervollständigen oder korrigieren.

Tab. 4.10: Use Case *capture group data*

Name	**search and select person**
Beschreibung	Aus der Liste der bereits in der ULR registrierten Personen wird eine Person ausgewählt.

Vorbedingung	Es ist mindestens eine Person in der ULR registriert.
Nachbedingung	Die gesuchte Person ist ausgewählt worden.
Hauptablauf	Dem Registrar wird eine einfache Suchmaske mit einem Eingabefeld gezeigt. Dort kann er seine Suchangaben eingeben. Dabei muss er nicht zwischen verschiedenen Feldern unterscheiden. Er kann z.B. Name und Vorname (oder auch andere Angaben) in beliebiger Reihenfolge eingeben. Nach dem Absetzen der Suche gibt die Applikation eine Liste der Benutzer aus, die den gemachten Suchangaben entsprechen. Aus dieser Liste wählt der Registrar die gesuchte Person aus.
alternativer Ablauf	keiner
Fehlerablauf	Werden mit den Suchangaben des Registrars keine Personen gefunden, teilt die Applikation dies dem Registrar mit einer Fehlermeldung mit und zeigt das Suchformular erneut an.

Tab. 4.11: Use Case *search and select person*

Name	**verify content**
Beschreibung	Die in den Use Cases *register single person* und *register group* gemachten Angaben werden verifiziert. Ausserdem werden diverse Bestätigungsdokumente ausgedruckt.
Vorbedingung	Die Use Cases *register single person* oder *register group* müssen ausgeführt worden sein.
Nachbedingung	Der Registrierungsstatus der Person ist auf "verifiziert" gesetzt.
Hauptablauf	Die Use Cases *print provisional document*, *correct/complete data*, *update status* und *print final document* werden nacheinander ausgeführt. Zwischen den einzelnen Use Cases kann problemlos eine längerer Zeitraum verstreichen, da unter Umständen auf Rückmeldungen gewartet werden muss.
alternativer Ablauf	Ist die Person bereits in einem externen Register erfasst und wurde dies in der ULR registriert (Use Case *create link to person registry*), wird auf die Use Cases *print provisional document*, *correct/complete data* und *update status* verzichtet. Nur der Use Case *print final document* wird noch ausgeführt.
Fehlerablauf	Kann die Identität der Person nicht verifiziert werden, wird dieser Use Case abgebrochen. Die Person erhält dann den Status "verifiziert" nicht.

Tab. 4.12: Use Case *verify content*

Name	**print provisional document**
Beschreibung	Ein provisorisches Dokument zur Verifizierung der Identität wird erstellt und ausgedruckt

Vorbedingung	keine weiteren
Nachbedingung	Das Dokument ist ausgedruckt und der Person zugeschickt worden.
Hauptablauf	Der Registrar druckt das von der Applikation generierte definitive Registrationsdokument aus, das alle bisher gemachten Angaben zusammenfasst. Dieses Dokument wird der Person per Post an die angegebene Adresse zugeschickt mit der Aufforderung, das Dokument zu korrigieren und zurückzuschicken.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.13: Use Case *print provisional document*

Name	**correct/complete data**
Beschreibung	Das zurückgeschickte Verifikationsdokument wird in die ULR registriert.
Vorbedingung	Das provisorische Registrationsdokument ist von der Person zurückgeschickt worden.
Nachbedingung	Die Korrekturen an der Registrierung sind gemacht.
Hauptablauf	Der Registrar durchsucht das zurückgeschickte Registrationsdokument nach Korrekturen. Findet er solche, trägt er sie in die ULR ein. Dazu benutzt er die Use Cases *capture person data* oder *capture group data*.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.14: Use Case *correct/complete data*

Name	**update status**
Beschreibung	Der Registrierungsstatus der Person wird auf “verifiziert“ gesetzt.
Vorbedingung	Das provisorische Registrationsdokument ist von der Person zurückgeschickt worden.
Nachbedingung	Der Registrierungsstatus der Person ist auf “verifiziert“ gesetzt.
Hauptablauf	Der Registrar stellt in dem von der Applikation ausgegebenen Formular den Registrierungsstatus der Person von “provisorisch“ auf “verifiziert“ und bestätigt diese Eingabe.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.15: Use Case *update status*

Name	**print final document**
Beschreibung	Es wird ein definitives Registrationsdokument zuhanden der registrierten Person ausgedruckt.
Vorbedingung	Der Registrierungsstatus der Person ist auf “verifiziert“ gesetzt.
Nachbedingung	Die Registrierung der Person ist mit diesem Schritt abgeschlossen.
Hauptablauf	Der Registrar druckt das von der Applikation generierte definitive Registrationsdokument aus und händigt es der Person aus. Das Dokument listet alle während der Registrierung gemachten Angaben der Person auf und dient der Person als Bestätigung ihrer Registrierung in der ULR.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.16: Use Case *print final document*

Zur Veranschaulichung dieser Use Cases wurden einige einfache Bildschirmmasken angefertigt, die zeigen sollen, wie sich der Registrar durch verschiedene von der ULR-Applikation generierte Formulare die notwendigen Informationen in die Datenbank eintragen kann. Die Bildschirmmasken sind in Abbildung 4.2 dargestellt.

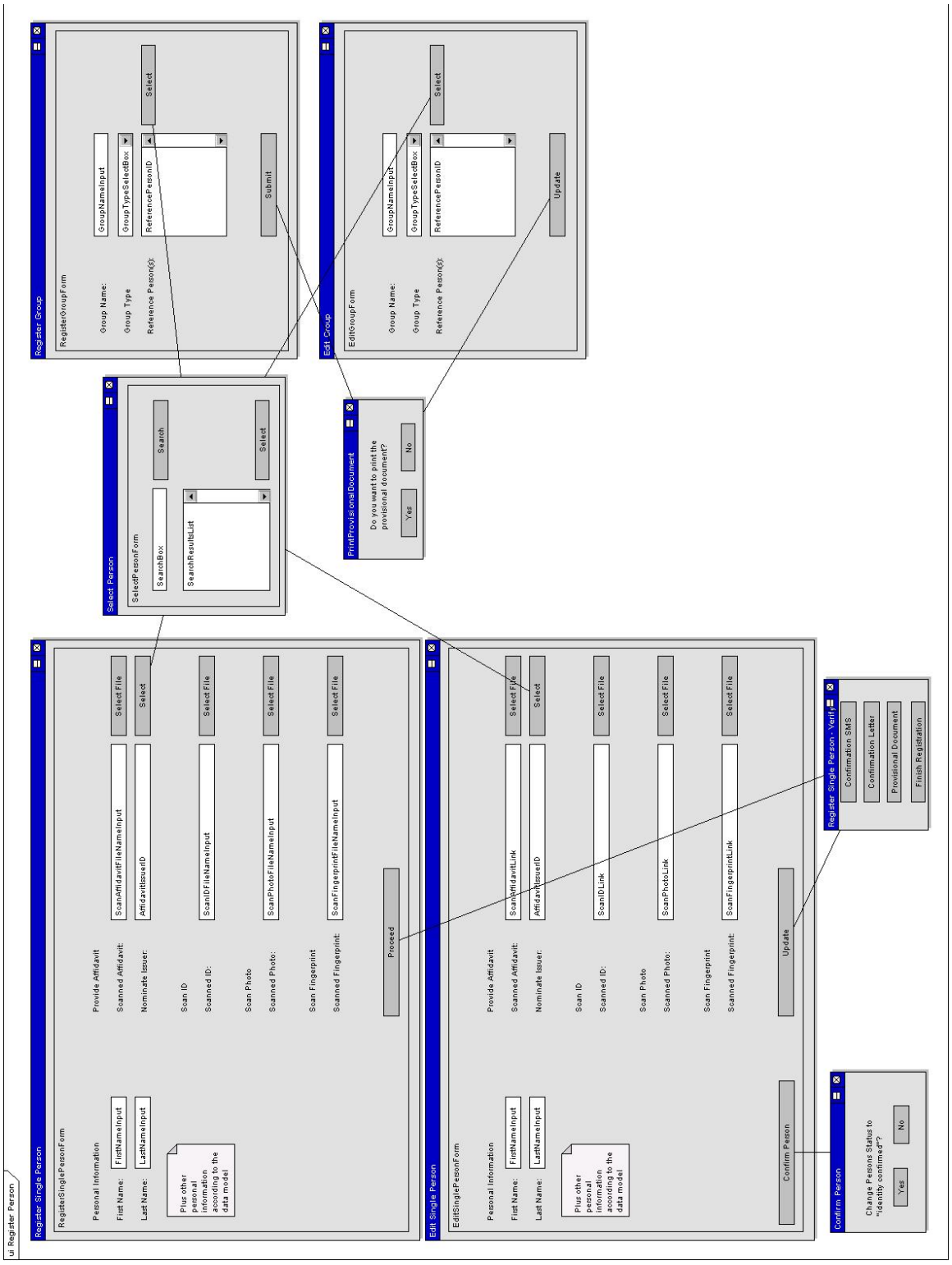


Abb. 4.2: Die Bildschirmmasken zur Registrierung von Personen

4.2.2 Landrechte registrieren

Diese Use Cases setzen den im Abschnitt 3.3 erwähnten Prozess der Registrierung von Landrechten um. Mit Rechten sind dabei nicht nur Besitzrechte sondern auch Nutzungsrechte gemeint. Eine Parzelle kann also auch mehrere Landrechte gleichzeitig aufweisen. Dies bedingt jedoch keine jeweils separaten Vorgehensweisen. Abbildung 4.3 zeigt die Use Cases in Zusammenhang mit der Registrierung von Landrechten. Nachfolgend werden die einzelnen Use Cases detailliert beschrieben. Use Cases, die bereits einmal beschrieben worden sind, werden hier nicht noch einmal besonders erwähnt:

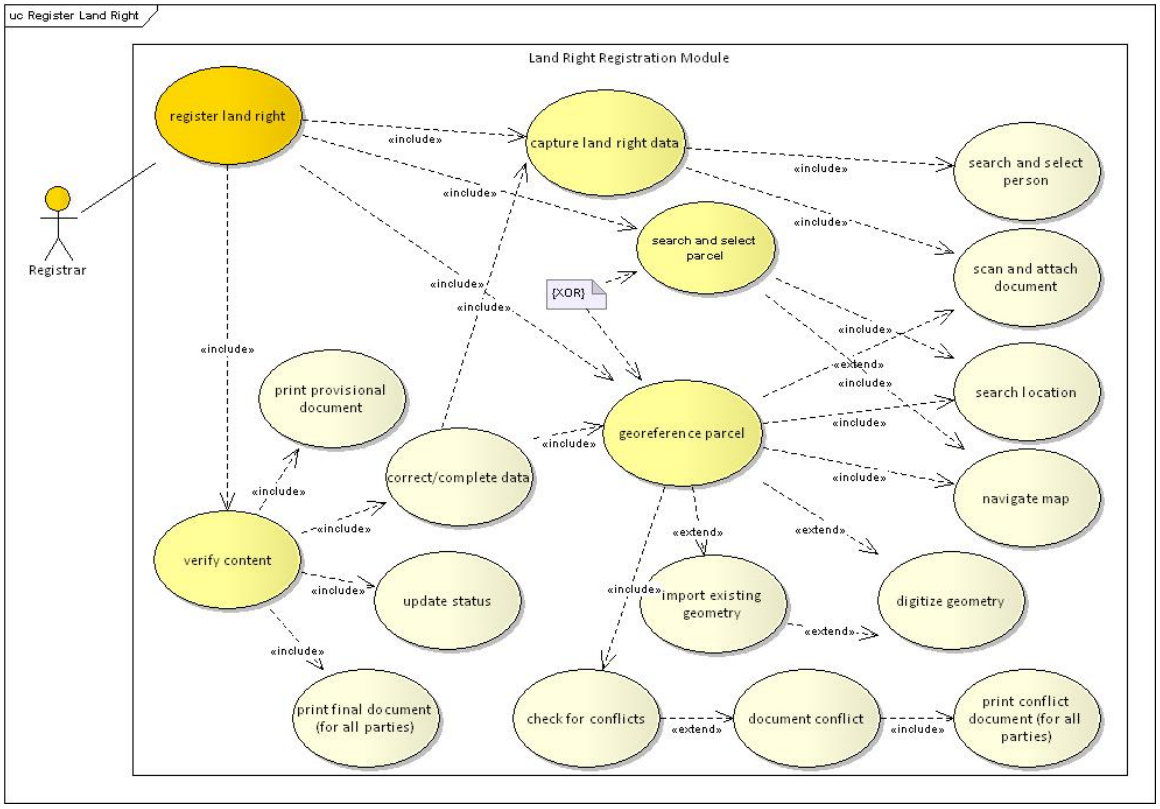


Abb. 4.3: Die Use Cases zur Registrierung von Landrechten

Name	**register land right**
Beschreibung	Dieser Use Case umfasst die vollständige Registrierung eines Landrechts in der ULR inkl. Georeferenzierung der Parzelle, Besitzererfassung, Konflikterkennung
Vorbedingung	Der Registrar ist korrekt am System angemeldet und hat die erforderlichen Rechte.

Nachbedingung	Das Landrecht ist vollständig (d.h. mit allen Angaben gemäss Datenmodell) in der ULR registriert und verifiziert.
Hauptablauf	Zunächst werden allgemeine Angaben zu dem Landrecht erfasst (Use Case *capture land right data*). Anschliessend wird die Parzelle georeferenziert (Use Case *georeference parcel*) inkl. Konflikterkennung und zum Schluss werden die Angaben verifiziert, Bestätigungsdokumente (Use Case *verify content*)ausgestellt.
alternativer Ablauf	Da die Parzelle bereits digitalisiert wurde, sucht der Registrar diese mit dem Use Case *search and select parcel*. Anschliessend führt er die Prüfung auf Konflikte aus (Use Case *check for conflicts*).
Fehlerablauf	keiner

Tab. 4.17: Use Case *register land right*

Name	**capture land right data**
Beschreibung	Allgemeine Angaben zum Landrecht (Besitzer, Dokumente, Art des Rechts) werden eingeholt und in die ULR geschrieben
Vorbedingung	Der Besitzer des Landrechts ist bereits in der ULR registriert. Es liegen Dokumente (offizieller oder informeller Art) vor, die den Besitzanspruch belegen.
Nachbedingung	Die allgemeinen Angaben zum Landrecht sind in der ULR gespeichert.
Hauptablauf	Der Registrar wählt mit dem Use Case *search and select person* den Besitzer des Landrechts aus. Danach fügt er mit dem Use Case *scan and attach document* eingescannte Dokumente hinzu, die den Besitzanspruch belegen.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.18: Use Case *capture land right data*

Name	**scan and attach document**
Beschreibung	Ein Scan des Dokuments wird gemacht und registriert.
Vorbedingung	Das Dokument liegt schriftlich vor und an der Workstation, in der die ULR-Applikation verwendet wird, ist ein Dokumentenscanner angeschlossen.
Nachbedingung	Der Scan des Dokuments ist in der ULR gespeichert.
Hauptablauf	Der Registrar scannt das Dokument mit dem Dokumentenscanner ein, so dass der Scan als Bilddatei vorliegt. Anschliessend wird die Bilddatei in das von der Applikation ausgegebene Formular eingetragen und in die ULR gespeichert.

alternativer Ablauf	keiner
Fehlerablauf	Ist die Bilddatei zu gross oder liegt sie nicht im korrekten Format vor, wird dem Registrar das Formular zusammen mit einer entsprechende Fehlermeldung erneut angezeigt.

Tab. 4.19: Use Case *scan and attach document*

Name	**georeference parcel**
Beschreibung	Das Landrecht wird im Raum verortet und gleichzeitig auf Konflikte mit anderen Landrechten geprüft.
Vorbedingung	Der Use Case *capture land right data* wurde erfolgreich ausgeführt.
Nachbedingung	Das Landrecht liegt in georeferenzierter Form vor.
Hauptablauf	Der Registrar digitalisiert die Parzelle nach Angaben des Besitzers in der ULR-Applikation (Use Cases *digitize geometry*, *search location* und *navigate map*). Danach fügt er existierende Dokumente (z.B. Kartenskizzen) mit dem Use Case *scan and attach document* in die ULR ein. Anschliessend führt er die Prüfung auf Konflikte aus (Use Case *check for conflicts*).
alternativer Ablauf 1	Der Registrar importiert eine bereits existierende Geometrie des Landrechts (z.B. in Form einer GPX-Datei; Use Case *import existing geometry*) in die ULR und editiert diese gegebenenfalls weiter (Use Case *digitize geometry*). Anschliessend führt er die Prüfung auf Konflikte aus (Use Case *check for conflicts*).
Fehlerablauf	keiner

Tab. 4.20: Use Case *georeference parcel*

Name	**search and select parcel**
Beschreibung	Aus der Menge der bereits digitalisierten Parzellen wird eine Parzelle ausgewählt.
Vorbedingung	Es ist mindestens eine Parzelle in der ULR registriert.
Nachbedingung	Die gesuchte Parzelle ist ausgewählt worden.
Hauptablauf	Die Applikation zeigt dem Registrar in einem Kartenfenster die in der ULR registrierten Parzellen an. Mittels der Use Cases *navigate map* und *search location* sucht der Registrar die gewünsche Parzelle. Hat er diese gefunden selektiert er sie mit dem Select-Werkzeug.

alternativer Ablauf	keiner
Fehlerablauf	Findet der Registrar die gesuchte Parzelle nicht, wird der Use Case abgebrochen und die Registrierung wird mit dem Use Case *georeference parcel* fortgesetzt.

Tab. 4.21: Use Case *search and select parcel*

Name	**digitize geometry**
Beschreibung	Die Geometrie des Landrechts wird am Bildschirm digitalisiert.
Vorbedingung	Der Registrar hat die ungefähre Position des Landrechts in der Bildschirmkarte lokalisiert (mit Hilfe der Use Cases *search location* und *navigate map*).
Nachbedingung	Die Geometrie des Landrechts ist in der ULR gespeichert.
Hauptablauf	Der Registrar digitalisiert die Geometrie des Landrechts am Bildschirm entweder als Punkt-, Linien- oder Flächeninformation. Er digitalisiert die Geometrie, indem er die einzelnen Vertices per Mausklick auf der Karte einträgt. Falsche oder überflüssige Vertices kann er verschieben oder löschen. Die Digitalisierung wird mit einem Doppelklick abgeschlossen. Für Punkte, Linien und Flächen stehen separate Digitalisierwerkzeuge zur Verfügung. Sind in der Region, in der digitalisiert wird, bereits andere Geometrien vorhanden, kann mittels Snapping (Fangen) ein lückenloser Anschluss an diese hergestellt werden.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.22: Use Case *digitize geometry*

Name	**import existing geometry**
Beschreibung	Bereits existierende Geometrien werden importiert.
Vorbedingung	Die Geometrie des Landrechts liegt digital vor (z.B. im GPX-Format). Sie wurde vorgängig vom Registrar zusammen mit dem Landbesitzer im Feld mit einem GPS-Gerät aufgenommen.
Nachbedingung	Die Geometrie ist in der ULR gespeichert
Hauptablauf	Der Registrar trägt die GPX-Datei in das von der Applikation ausgegebene Formular ein. Die Applikation parst die Datei und zeigt die Geometrie auf dem Bildschirm an. Anschliessend wird die Geometrie korrigiert (Use Case *digitize geometry*).

alternativer Ablauf	Gegebenenfalls kann auf die Korrektur der importierten Geometrie verzichtet werden.
Fehlerablauf	Enthält die GPX-Datei Fehler, werden diese dem Registrar angezeigt und das Formular zur Dateiangabe erneut angezeigt.

Tab. 4.23: Use Case *import existing geometry*

Name	**search location**
Beschreibung	Navigationshilfe durch einfache Ortssuche
Vorbedingung	Es sind durchsuchbare Basisdaten vorhanden.
Nachbedingung	Der Kartenausschnitt ist auf den gesuchten Ort gezoomt.
Hauptablauf	Dem Registrar wird eine einfache Suchmaske mit einem Eingabefeld gezeigt. Dort kann er seine Suchangaben eingeben. Dabei muss er nicht zwischen verschiedenen Feldern unterscheiden. Nach dem Absetzen der Suche gibt die Applikation eine Liste der Orte aus, die den gemachten Suchangaben entsprechen. Aus dieser Liste wählt der Registrar den gesuchten Ort aus. Danach verschiebt die Applikation den Kartenausschnitt auf diesen Ort.
alternativer Ablauf	keiner
Fehlerablauf	Werden mit den Suchangaben des Registrars keine Orte gefunden, teilt die Applikation dies dem Registrar mit einer Fehlermeldung mit und zeigt das Suchformular erneut an.

Tab. 4.24: Use Case *search location*

Name	**navigate map**
Beschreibung	Ermöglicht dem Anwender die einfache Navigation in der Karte.
Vorbedingung	keine
Nachbedingung	keine
Hauptablauf	Dem Anwender werden die bekannten Navigationswerkzeuge so dargestellt, dass er sie sofort als solche erkennt und benutzen kann. Konkret handelt es sich um Zoom-Werkzeuge (Zoom In, Zoom Out), Pan-Werkzeuge (Verschieben des Kartenausschnitts) oder um das Layermanagement (Ein- und Ausschalten von Layern). Diese Werkzeuge sind weit verbreitet und bekannt, so dass sie hier nicht näher beschrieben werden müssen.

alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.25: Use Case *navigate map*

Name	**check for conflicts**
Beschreibung	Prüfung auf sich überlappende Landrechte.
Vorbedingung	Das Landrecht ist bereits georeferenziert und es liegen in der gleichen Region bereits andere georeferenzierte Landrechte vor.
Nachbedingung	Der Konflikt ist erkannt und dokumentiert. Der Konfliktstatus des Landrechts steht entweder auf "umstritten" oder "unbestritten".
Hauptablauf	Der Registrar löst die Prüfung mit einem Klick auf den entsprechenden Knopf in der Applikation aus. Die Applikation prüft, ob in der ULR bereits Landrechte vorliegen, die mit dem jetzt digitalisierten Landrecht überlappen (eine entsprechende Buffergrösse kann spezifiziert werden). Es liegen Konflikte vor, die Applikation geht zum Use Case *document conflict über*. Der Registrationsstatus des Landrechts wird auf "umstritten" gesetzt.
alternativer Ablauf	Der Registrar löst die Prüfung mit einem Klick auf den entsprechenden Knopf in der Applikation aus. Die Applikation prüft, ob in der ULR bereits Landrechte vorliegen, die mit dem jetzt digitalisierten Landrecht überlappen (eine entsprechende Buffergrösse kann spezifiziert werden). Es liegen keine Konflikte vor, der Registrationsstatus des Landrechts wird auf "unbestritten" gesetzt.
Fehlerablauf	keiner

Tab. 4.26: Use Case *check for conflicts*

Name	**document conflict**
Beschreibung	Der Landkonflikt wird dokumentiert.
Vorbedingung	Es ist ein Konfliktfall aufgetreten.
Nachbedingung	Der Konflikt ist mit Zusatzangaben versehen worden und in der ULR abgespeichert.
Hauptablauf	Die Applikation stellt den Konflikt in einer Karte am Bildschirm dar und zeigt gleichzeitig ein Formular für Zusatzinformationen zum Konflikt (erste Einschätzung der Lage, Höhe des Konfliktpotenzials etc.) an. Der Registrar füllt dieses Formular aus. Die Angaben werden in der ULR gespeichert.

alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.27: Use Case *document conflict*

Name	**print conflict document**
Beschreibung	Die involvierten Personen werden mit der Dokumentation ausgestattet.
Vorbedingung	Der Landkonflikt ist dokumentiert.
Nachbedingung	Alle involvierten Personen verfügen über die Dokumentation des Konfklikts.
Hauptablauf	Die Applikation generiert automatisch ein Dokument mit allen relevanten Informationen zu dem Landkonflikt (Kartenausschnitte, Angaben über die Besitzer, Art des Landrechts etc.) aus der ULR. Der Registrar druckt dieses Dokument aus und stellt es den involvierten Personen zu (Landrechtsbesitzer, Schlichtungsstelle etc.).
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.28: Use Case *print conflict document*

Zur Veranschaulichung dieser Use Cases wurden einige einfache Bildschirmmasken angefertigt, die zeigen sollen, wie sich der Registrar durch verschiedene von der ULR-Applikation generierte Formulare die notwendigen Informationen in die Datenbank eintragen kann. Die Bildschirmmasken sind in Abbildung 4.4 dargestellt.

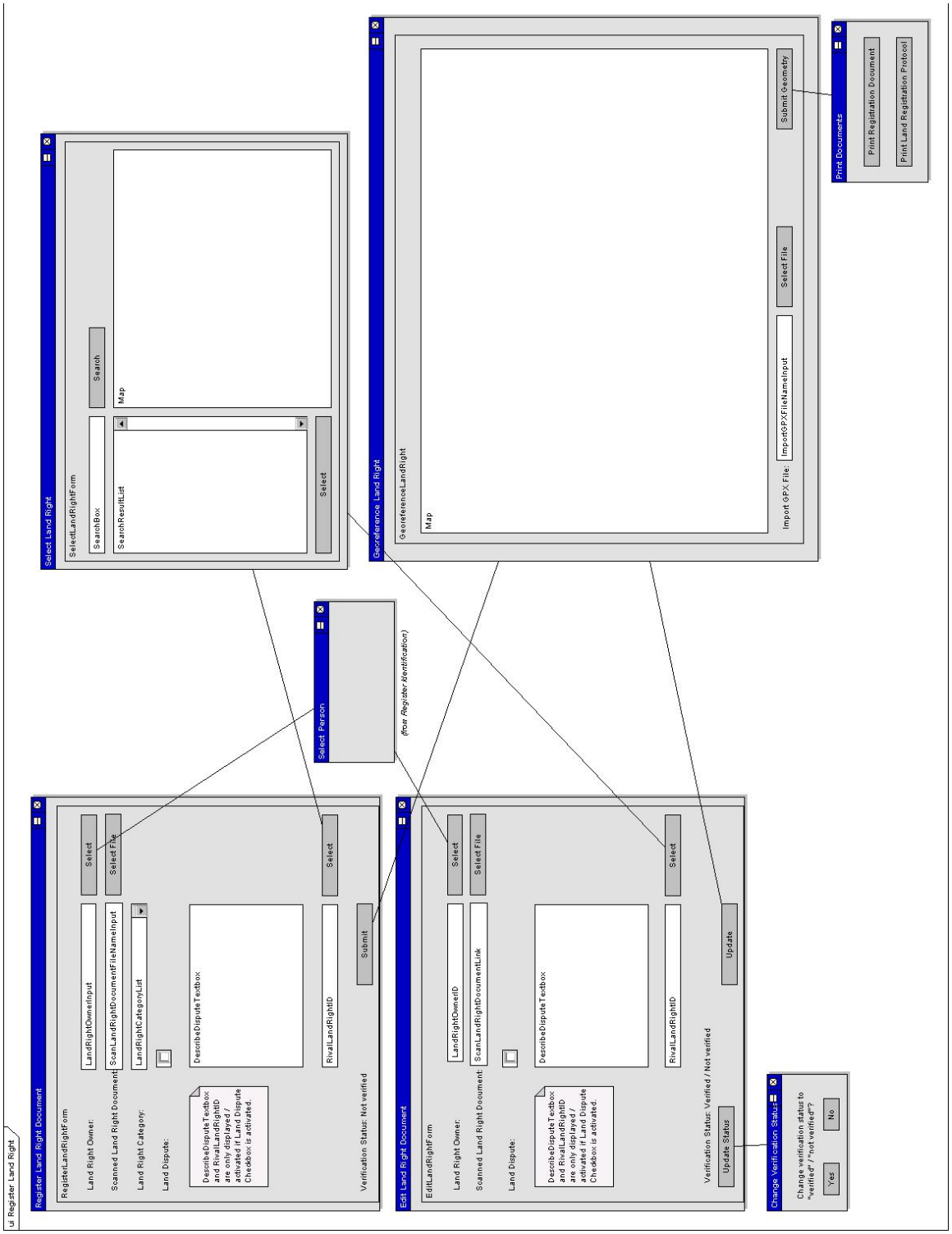


Abb. 4.4: Die Bildschirmmasken zur Registrierung von Landrechten

4.2.3 Basisdaten registrieren

Diese Use Cases ergänzen die ULR mit für die Georeferenzierung wichtigen Basisdaten. Basisdaten sind Geodaten (Vektor oder Raster), die dem Registrar bei der Georeferenzierung entweder als Digitalisierungshintergrund (Luftbilder) oder als Orientierungshilfen dienen (Points of Interest, Landmarks, Strassen, andere topographische Elemente). Die Bandbreite ist sehr gross. Deshalb können diese Use Cases auch nicht alle denkbaren Fälle abdecken. Hier muss die Applikation mit der Zeit wachsen. Es ist auch nicht sehr sinnvoll, wenn in der ULR grosse Mengen an solchen Basisdaten abgelegt werden. Solche Daten sollen mit der Zeit direkt aus der NGDI Tansanias bezogen werden. Bis eine solche NGDI jedoch existiert, wird übergangsmässig mit einer Reihe von externen Datenanbietern gearbeitet. Abbildung 4.5 zeigt die involvierten Use Cases im Überblick. Nachfolgend werden die einzelnen Use Cases detailliert beschrieben.

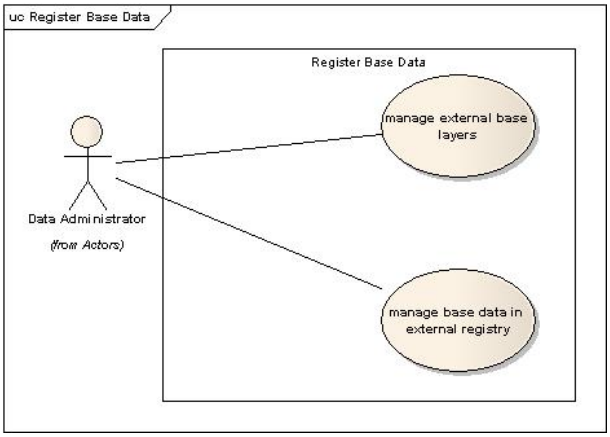


Abb. 4.5: Die Use Cases zur Registrierung von Basisdaten

Name	**manage external base layers**
Beschreibung	Verwaltung der extern abgelegten Basisdaten.
Vorbedingung	Der Data Administrator ist korrekt am System angemeldet und hat die erforderlichen Rechte.
Nachbedingung	Die benötigten Basislayer sind korrekt konfiguriert.
Hauptablauf	Der Data Administrator kann mit diesem Interface vordefinierte Basislayer (Google Maps, Yahoo Maps, Microsoft Virtual Earth, Openstreetmap, Openaerialmap etc.) für den Gebrauch in der ULR-Applikation freischalten. Ist ein solcher Basislayer aktiviert, ist er für den Anwender in der ULR-Applikation sichtbar. Ist er deaktiviert, ist er für den Anwender nicht sichtbar. Neben den vordefinierten Basislayern kann der Data Administrator auch beliebige weitere Layer über standardisierte Schnittstellen (WMS, WFS) einbinden und dem Anwender zur Verfügung stellen.

alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.29: Use Case *manage external base layers*

Name	**manage base data in external registry**
Beschreibung	Editieren von Basisdaten in ausgewählten externen Diensten.
Vorbedingung	Der Data Administrator verfügt über das notwendige Know-How.
Nachbedingung	keine
Hauptablauf	Der Data Administrator kann bestimmte externe Basislayer editieren. Beispiele dafür sind beispielsweise Openstreetmap, Openaerialmap oder Geonames. In diesen Diensten kann ein beliebiger Anwender Daten digitalisieren und editieren. Um die Datenmenge in der ULR zu limitieren, wird der Data Administrator Basisdaten, die er selber editieren oder digitalisieren möchte, in einem der externen Dienste bearbeiten und dann in die ULR-Applikation einbinden (Use Case *manage external base layers*).
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.30: Use Case *manage base data in external registry*

4.3 Transaktionen

Die folgenden Use Case beschreiben die Abwicklung einer Transaktion. Dabei wird ein Landrecht von einer Person auf eine andere übertragen. Abbildung 4.6 zeigt die beteiligten Use Cases. Nachfolgend werden die einzelnen Use Cases detailliert beschrieben. Use Cases, die bereits einmal beschrieben worden sind, werden hier nicht noch einmal besonders erwähnt:

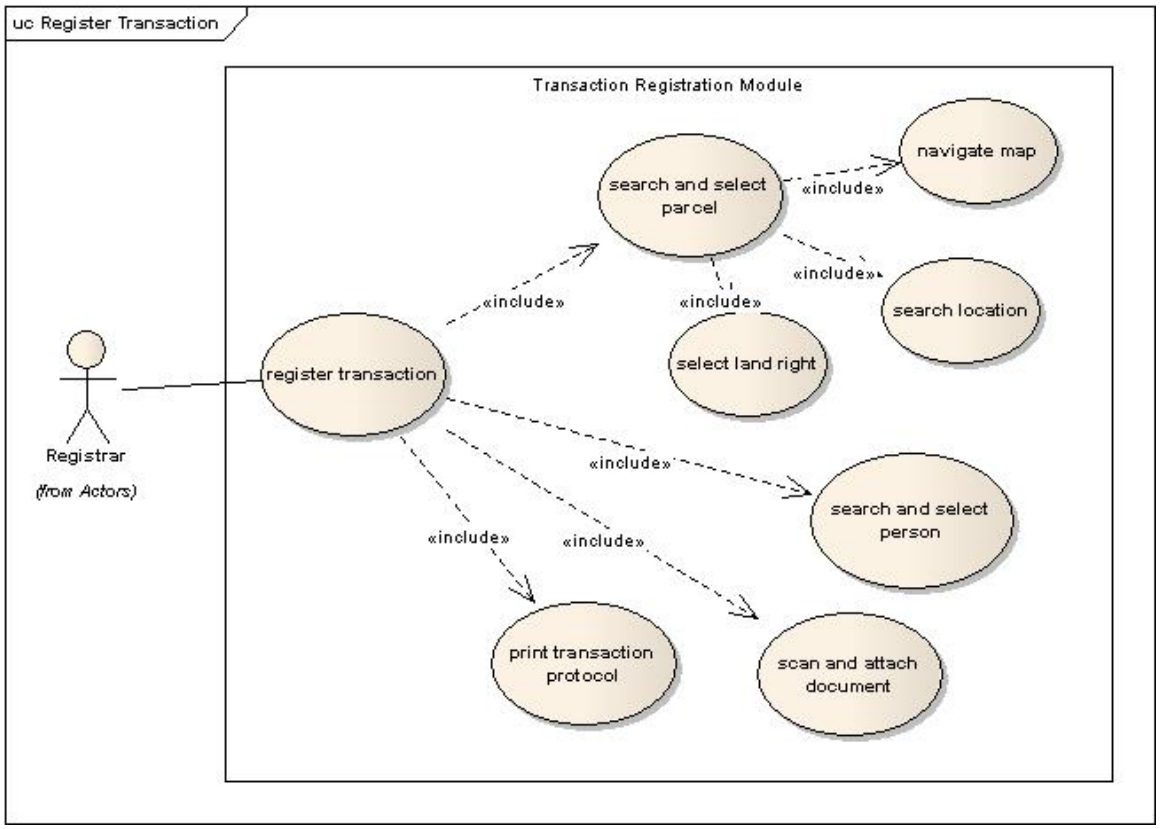


Abb. 4.6: Die Use Cases zur Registrierung von Transaktionen

Name	**register transaction**
Beschreibung	Registriert einen Wechsel eines Landrechts von einer Person auf eine andere.
Vorbedingung	Der neue Besitzer, das betroffene Landrecht und der alte Besitzer müssen in der ULR registriert sein.
Nachbedingung	Das Landrecht ist auf den neuen Besitzer lautend in der ULR gespeichert.

Hauptablauf	Der Registrar sucht mit dem Use Case *search and select parcel* die Parzelle, für die das Landrecht gilt. Daraufhin wählt der Registrar mit dem Use Case *select land right* das zu transferierende Landrecht aus. Danach wählt er mit dem Use Case *search and select person* den neuen Besitzer des Landrechts aus. Allfällige Transaktionsdokumente (Verkaufsurkunden etc.) werden mit dem Use Case *scan and attach document* in die ULR gespeichert. Abschliessend wird mit dem Use Case *print transaction protocol* eine Bestätigung der Transaktion ausgedruckt.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.31: Use Case *register transaction*

Name	**select land right**
Beschreibung	Auswahl des gesuchten Landrechts.
Vorbedingung	Eine Parzelle wurde mit dem Use Case *search and select parcel* ausgewählt.
Nachbedingung	Das gesuchte Landrecht wurde ausgewählt.
Hauptablauf	Die Applikation generiert eine Auswahlliste mit allen Landrechten, die auf der zuvor ausgewählten Parzelle eingetragen sind. Der Registrar wählt aus dieser Liste das gesuchte Landrecht aus.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.32: Use Case *select land right*

Name	**print transaction protocol**
Beschreibung	Generierung eines Transaktionsprotokolls.
Vorbedingung	Die Transaktion wurde erfolgreich registriert.
Nachbedingung	Das Protokoll wurde ausgedruckt.
Hauptablauf	Der Registrar löst die Generierung eines Transaktionsprotokolls aus, in dem alle Informationen zu der Transaktion festgehalten sind. Er druckt dieses aus und händigt es allen involvierten Personen aus (alter Besitzer, neuer Besitzer).
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.33: Use Case *print transaction protocol*

Zur Veranschaulichung dieser Use Cases wurden einige einfache Bildschirmmasken angefertigt, die zeigen sollen, wie sich der Registrar durch verschiedene von der ULR-Applikation generierte Formulare die notwendigen Informationen in die Datenbank eintragen kann. Die Bildschirmmasken sind in Abbildung 4.7 dargestellt.

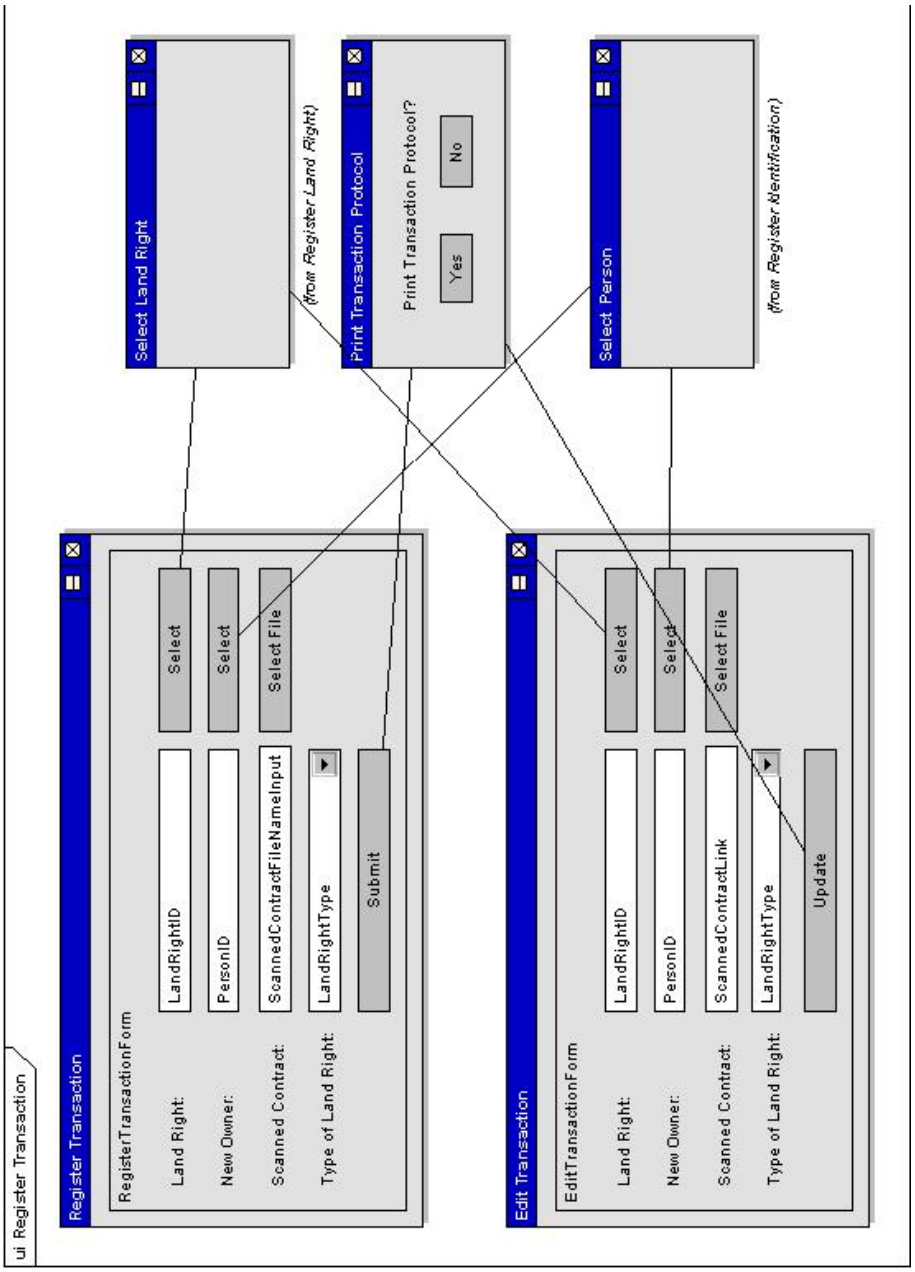


Abb. 4.7: Die Bildschirmmasken zur Registrierung von Transaktionen

4.4 Registry abfragen

Die Inhalte der ULR sollen auch betrachtet werden können. Diese Funktionalität wird den folgenden Use Cases (s. Abbildung 4.8) abgedeckt. Nachfolgend werden die einzelnen Use Cases detailliert beschrieben. Use Cases, die bereits einmal beschrieben worden sind, werden hier nicht noch einmal besonders erwähnt:

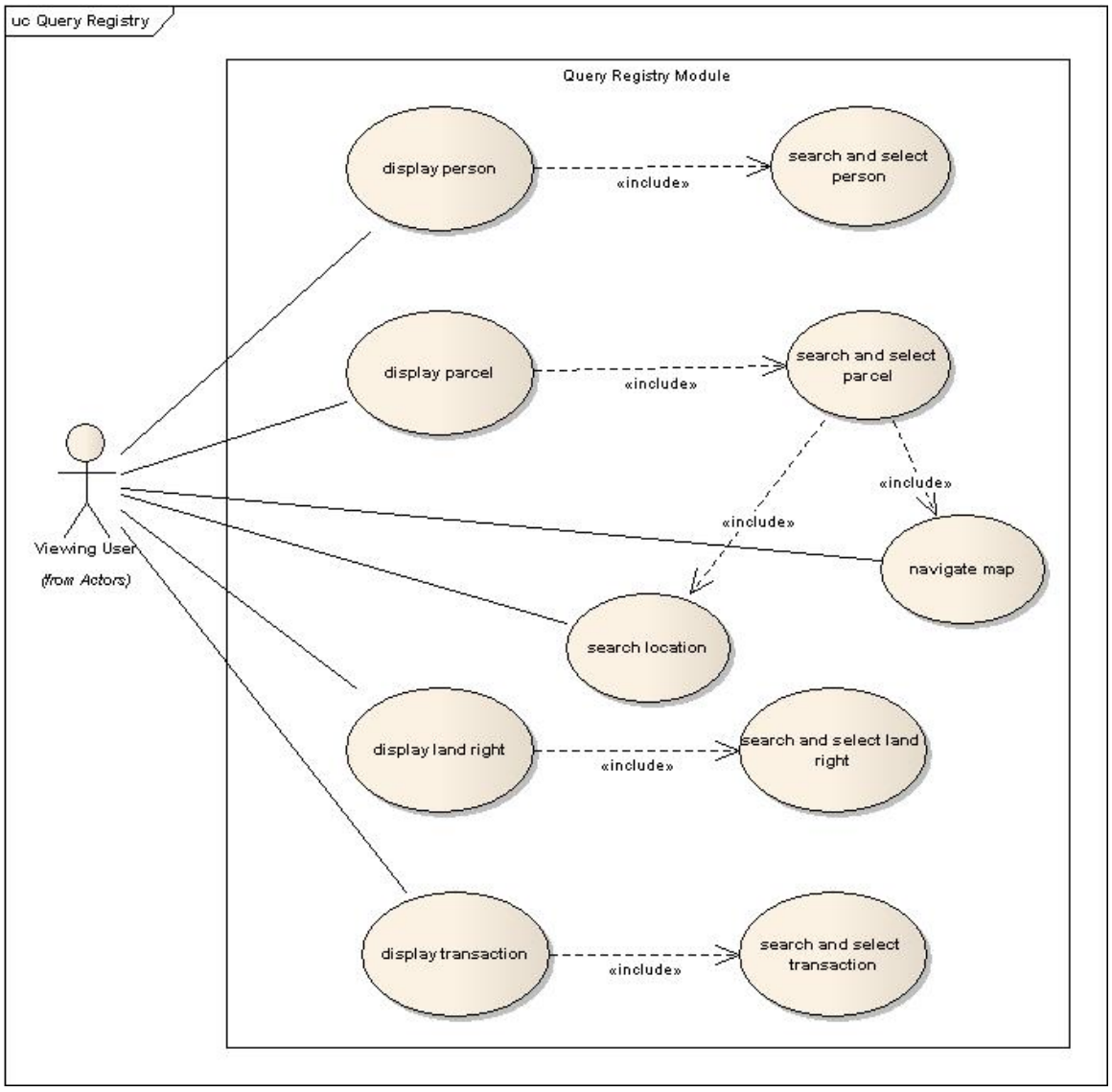


Abb. 4.8: Die Use Cases zum Betrachten der ULR

Name	**display land right**
Beschreibung	Alle Informationen eines einzelnen Landrechtes werden angezeigt.
Vorbedingung	keine
Nachbedingung	Das Landrecht wird angezeigt.
Hauptablauf	Der Betrachter wählt mit dem Use Case *search and select land right* das gesuchte Landrecht aus. Anschliessend zeigt die Applikation dem Anwender alle Informationen zu dem gewählten Landrecht (inkl. Kartendarstellung) an.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.34: Use Case *display land right*

Name	**search and select land right**
Beschreibung	Aus der Liste der bereits in der ULR registrierten Landrechte wird ein Landrecht ausgewählt.
Vorbedingung	Es ist mindestens ein Landrecht in der ULR registriert.
Nachbedingung	Das gesuchte Landrecht ist ausgewählt worden.
Hauptablauf	Dem Registrar wird eine einfache Suchmaske mit einem Eingabefeld gezeigt. Dort kann er seine Suchangaben eingeben. Dabei muss er nicht zwischen verschiedenen Feldern unterscheiden. Er kann z.B. Name und Vorname (oder auch andere Angaben) in beliebiger Reihenfolge eingeben. Nach dem Absetzen der Suche gibt die Applikation eine Liste der Landrechte aus, die den gemachten Suchangaben entsprechen. Aus dieser Liste wählt der Registrar die gesuchte Person aus.
alternativer Ablauf	keiner
Fehlerablauf	Werden mit den Suchangaben des Registrars keine Landrechte gefunden, teilt die Applikation dies dem Registrar mit einer Fehlermeldung mit und zeigt das Suchformular erneut an.

Tab. 4.35: Use Case *search and select land right*

Name	**display transaction**
Beschreibung	Alle Informationen einer einzelnen Transaktion werden angezeigt.
Vorbedingung	keine
Nachbedingung	Die Transaktion wird angezeigt.
Hauptablauf	Der Betrachter wählt mit dem Use Case *search and select land transaction* die gesuchte Transaktion aus. Anschliessend zeigt die Applikation dem Anwender alle Informationen zur gewählten Transaktion (inkl. Kartendarstellung) an.

alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.36: Use Case *display transaction*

Name	**search and select transaction**
Beschreibung	Aus der Liste der bereits in der ULR registrierten Transaktionen wird eine Transaktion ausgewählt.
Vorbedingung	Es ist mindestens eine Transaktion in der ULR registriert.
Nachbedingung	Die gesuchte Transaktion ist ausgewählt worden.
Hauptablauf	Dem Registrar wird eine einfache Suchmaske mit einem Eingabefeld gezeigt. Dort kann er seine Suchangaben eingeben. Dabei muss er nicht zwischen verschiedenen Feldern unterscheiden. Er kann z.B. das Datum der Transaktion (oder auch andere Angaben) in beliebiger Reihenfolge eingeben. Nach dem Absetzen der Suche gibt die Applikation eine Liste der Transaktionen aus, die den gemachten Suchangaben entsprechen. Aus dieser Liste wählt der Registrar die gesuchte Transaktion aus.
alternativer Ablauf	keiner
Fehlerablauf	Werden mit den Suchangaben des Registrars keine Transaktionen gefunden, teilt die Applikation dies dem Registrar mit einer Fehlermeldung mit und zeigt das Suchformular erneut an.

Tab. 4.37: Use Case *search and select transaction*

Name	**display person**
Beschreibung	Alle Informationen einer einzelnen Person werden angezeigt.
Vorbedingung	keine
Nachbedingung	Die Person wird angezeigt.
Hauptablauf	Der Betrachter wählt mit dem Use Case *search and select person* die gesuchte Person aus. Anschliessend zeigt die Applikation dem Anwender alle Informationen zu der gewählten Person an.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.38: Use Case *display person*

Name	**display parcel**
Beschreibung	Alle Informationen einer einzelnen Parzelle werden angezeigt.
Vorbedingung	keine
Nachbedingung	Die Parzelle wird angezeigt.

Hauptablauf	Der Betrachter wählt mit dem Use Case *search and select person* die gesuchte Parzelle aus. Anschliessend zeigt die Applikation dem Anwender alle Informationen zur gewählten Parzelle (inkl. Kartendarstellung) an.
alternativer Ablauf	keiner
Fehlerablauf	keiner

Tab. 4.39: Use Case *display parcel*

4.5 Systemadministration

Mit diesen Use Cases (s. Abbildung 4.9) kümmert sich der System Administrator um die Verwaltung der lokalen ULR-Applikation. Darin inbegriffen ist die Benutzerverwaltung sowie die Steuerung der Synchronisation und Replikation zu den anderen Knoten im verteilten ULR-System (s. Abschnitt 3.2). Nachfolgend werden die einzelnen Use Cases detailliert beschrieben.

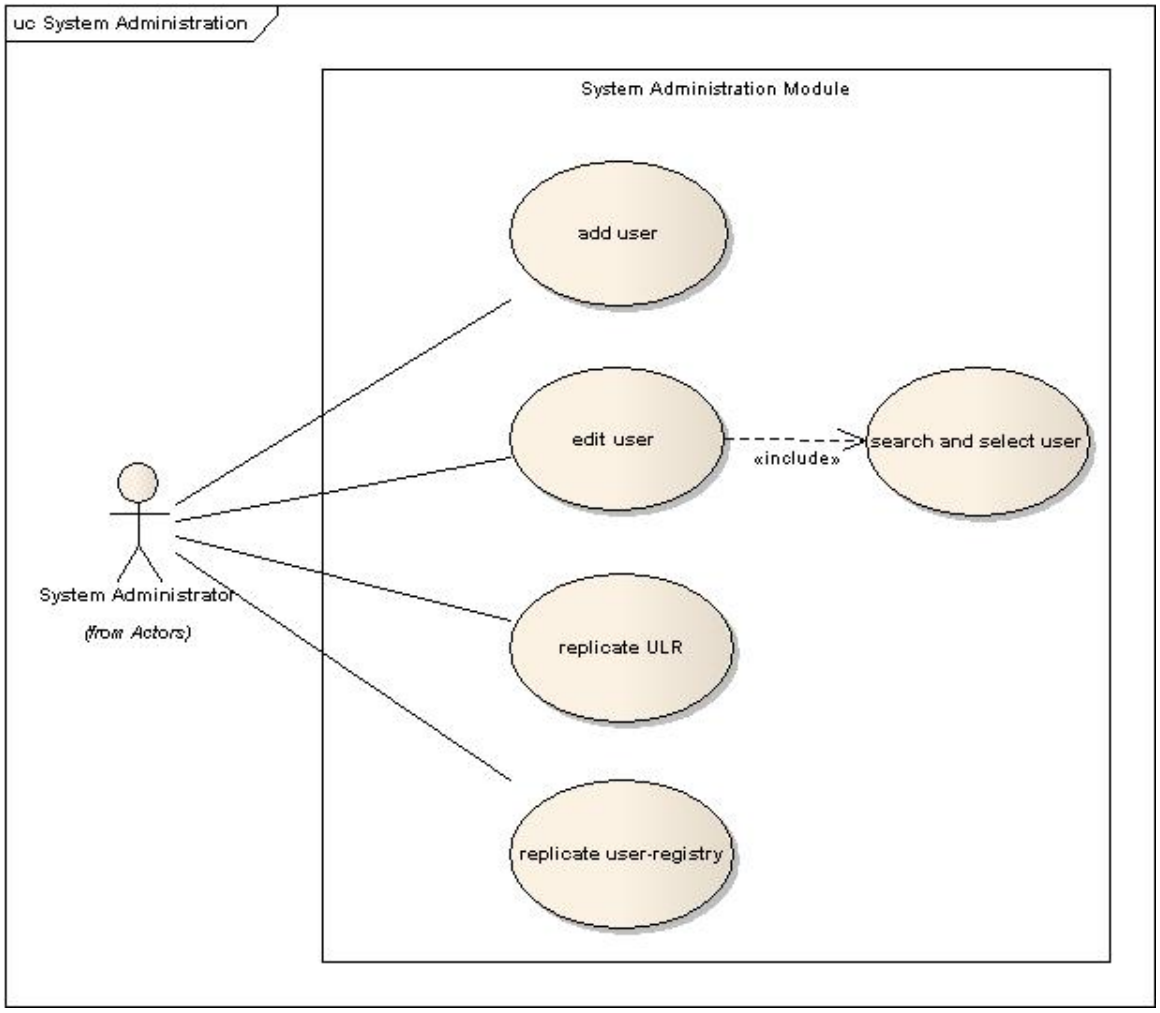


Abb. 4.9: Die Use Cases zur Systemverwaltung

Name	**add user**
Beschreibung	Ein neuer Benutzer wird der Benutzerverwaltung hinzugefügt.
Vorbedingung	Der System Administrator ist am System angemeldet und verfügt über die notwendigen Rechte.
Nachbedingung	Der neue Benutzer ist in der Benutzerverwaltung registriert.
Hauptablauf	Die Applikation generiert ein Formular, in das der System Administrator die Angaben zu dem Benutzer einträgt (Benutzername, Passwort, richtiger Name, Rolle). Die Applikation überprüft die gemachten Angaben und schreibt sie in die Benutzerverwaltung.
alternativer Ablauf	keiner
Fehlerablauf	Sind die vom System Administrator gemachten Angaben nicht korrekt oder unvollständig, gibt die Applikation eine Fehlermeldung aus und zeigt das Formular erneut zur Eingabe an.

Tab. 4.40: Use Case *add user*

Name	**edit user**
Beschreibung	Editiert die Angaben zu einem bereits registrierten Benutzer.
Vorbedingung	Der System Administrator ist am System angemeldet und verfügt über die notwendigen Rechte. Der Benutzer ist bereits in der Benutzerverwaltung registriert.
Nachbedingung	Der Benutzer wurde entsprechend abgeändert.
Hauptablauf	Mit dem Use Case *search and select user* wählt der System Administrator den gesuchten Benutzer aus. Die Applikation zeigt in einem Formular die existierenden Angaben zu dem Benutzer an. Der System Administrator kann diese Angaben nun editieren. Die Applikation überprüft die gemachten Angaben und schreibt sie in die Benutzerverwaltung zurück.
alternativer Ablauf	keiner
Fehlerablauf	Sind die vom System Administrator gemachten Angaben nicht korrekt oder unvollständig, gibt die Applikation eine Fehlermeldung aus und zeigt das Formular erneut zur Eingabe an.

Tab. 4.41: Use Case *edit user*

Name	search and select user
Beschreibung	Ein bestimmter Benutzer wird in der Benutzerverwaltung gesucht.
Vorbedingung	Der Benutzer ist bereits in der Benutzerverwaltung registriert.
Nachbedingung	Der gesuchte Benutzer wurde ausgewählt.
Hauptablauf	Dem System Administrator wird eine einfache Suchmaske mit einem Eingabefeld gezeigt. Dort kann er seine Suchangaben eingeben. Dabei muss er nicht zwischen verschiedenen Feldern unterscheiden. Er kann z.B. Benutzername und Rolle (oder auch andere Angaben) in beliebiger Reihenfolge eingeben. Nach dem Absetzen der Suche gibt die Applikation eine Liste der Benutzer aus, die den gemachten Suchangaben entsprechen. Aus dieser Liste wählt der System Administrator den gesuchten Benutzer aus.
alternativer Ablauf	keiner
Fehlerablauf	Werden mit den Suchangaben des System Administrators keine Benutzer gefunden, teilt die Applikation dies dem System Administrator mit einer Fehlermeldung mit und zeigt das Suchformular erneut an.

Tab. 4.42: Use Case *search and select user*

Name	replicate ULR
Beschreibung	Synchronisiert die ULR mit anderen ULR-Knoten.
Vorbedingung	Der System Administrator ist am System angemeldet und verfügt über die notwendigen Rechte. Der lokale ULR-Knoten kann mit dem Ziel-Knoten Verbindung aufnehmen (Internet, Intranet etc.).
Nachbedingung	Der lokale ULR-Knoten ist auf dem gleichen Stand wie der Ziel-Knoten.
Hauptablauf	Die Applikation zeigt dem System Administrator eine Liste mit verfügbaren Ziel-Knoten an. Diese können in der verteilten Hierarchie höher oder auch tiefer als der lokale Knoten sein. Der System Administrator wählt den gewünschten Ziel-Knoten aus und startet die Synchronisation. Nach Abschluss der Synchronisation zeigt die Applikation dem System Administrator eine Log-Datei über die getätigte Synchronisation an.
alternativer Ablauf	keiner
Fehlerablauf	Falls die Synchronisation fehlschlägt, zeigt die Applikation dies über eine Log-Datei an.

Tab. 4.43: Use Case *replicate ULR*

Name	**replicate user-registry**
Beschreibung	Synchronisiert die Benutzerverwaltung mit anderen Knoten
Vorbedingung	Der System Administrator ist am System angemeldet und verfügt über die notwendigen Rechte. Der lokale Knoten kann mit dem Ziel-Knoten Verbindung aufnehmen (Internet, Intranet etc.).
Nachbedingung	Die lokale Benutzerverwaltung ist auf dem gleichen Stand wie die des Ziel-Knotens.
Hauptablauf	Die Applikation zeigt dem System Administrator eine Liste mit verfügbaren Ziel-Knoten an. Diese können in der verteilten Hierarchie höher oder auch tiefer als der lokale Knoten sein. Der System Administrator wählt den gewünschten Ziel-Knoten aus und startet die Synchronisation. Nach Abschluss der Synchronisation zeigt die Applikation dem System Administrator eine Log-Datei über die getätigte Synchronisation an.
alternativer Ablauf	keiner
Fehlerablauf	Falls die Synchronisation fehlschlägt, zeigt die Applikation dies über eine Log-Datei an.

Tab. 4.44: Use Case *replicate user-registry*

5 Logische Systemarchitektur

Die Anforderungen (Kapitel 3) und die Use Cases (Kapitel 4) bilden zusammen die funktionalen Anforderungen an die ULR-Applikation. Sie werden in logischen Einheiten zusammengefasst und in einem UML-Komponentendiagramm abgebildet (s. Abbildung 5.1 oben). Im Diagramm sind neben den logischen Einheiten auch die notwendigen Interfaces dargestellt, die die Kommunikation zwischen den Modulen ermöglichen. Das Diagramm stellt die logische Systemarchitektur der ULR-Applikation dar. Diese logische Systemarchitektur folgt dem 3-Schichten-Ansatz (RÄTZMANN 1998; WIKIPEDIA 2008).

Die unterste Schicht umfasst die Datenhaltung. Dies ist einerseits eine objekt-relationale Datenbank, die den SFS-Standard unterstützt. Daneben steht ein AAA-Dienst (Authentifizierung, Autorisierung und Abrechnung), der für die Benutzerverwaltung und die Sicherheit zuständig ist. Der Dienst verwaltet die Benutzer und deren Berechtigungen und prüft auch, für welche Funktionen ein Benutzer berechtigt ist. Ausserdem ist hier ein Integrationswerkzeug für Geodaten vorgesehen. Damit ist ein Mechanismus gemeint, wie bestehende Geodaten (Luftbilder, freie Vektordaten etc.) direkt in die Datenbank eingespielt werden können. Zur untersten Schicht gehören auch allfällige ULR-Knoten, die unterhalb des aktuellen Knotens liegen. Deren Daten werden im aktuellen Knoten über einen WFS- oder einen WMS-Dienst eingebunden und verfügbar gemacht.

In der mittleren Schicht (Applikationsschicht) befindet sich ein Webserver, der drei Komponenten beherbergt. Zum Einen ein Geo-Applikationsframework, das Geodaten aufbereiten und via standardisierte Schnittstellen (OGC-Standards) bereitstellen kann. Zum Anderen ein konventionelles Webframework, das die eigentliche Webapplikation in Form von HTML-Seiten bereitstellt. Dazu kommt ein Replikationsdienst, der die Synchronisation der Benutzerverwaltung (des AAA-Dienstes) des lokalen Knotens mit anderen ULR-Knoten auf einem höheren oder tieferen Hierarchielevel der verteilten Architektur (s. Abschnitt 3.2) steuert. Die Kommunikation zwischen Applikationsschicht und Datenschicht erfolgt über WMS, WFS, ODBC/JDBC oder über das LDAP-Protokoll. Auch die Synchronisation der Benutzerverwaltung erfolgt über das LDAP-Protokoll.

In der obersten Schicht befindet sich die Client-Anwendung (Präsentationsschicht). Sie ist eine Web-Applikation, die in einem beliebigen modernen Webbrowser lauffähig ist und besteht aus sechs Modulen, in denen die Funktionen gekapselt sind, die mit den Use Cases im vorherigen Kapitel beschrieben worden sind. Neben der Web-Applikation können auch übergeordnete ULR-Knoten (gemäss der verteilten Architektur) stehen. Diese übergeordneten Knoten binden die Daten des lokalen Knotens direkt ein. Daneben können auch andere GIS-Clients eingesetzt werden, die eine der Schnittstellen (WFS oder WMS) nutzen können. Zwischen der Client und der Middleware-Schicht wird mittels WMS und

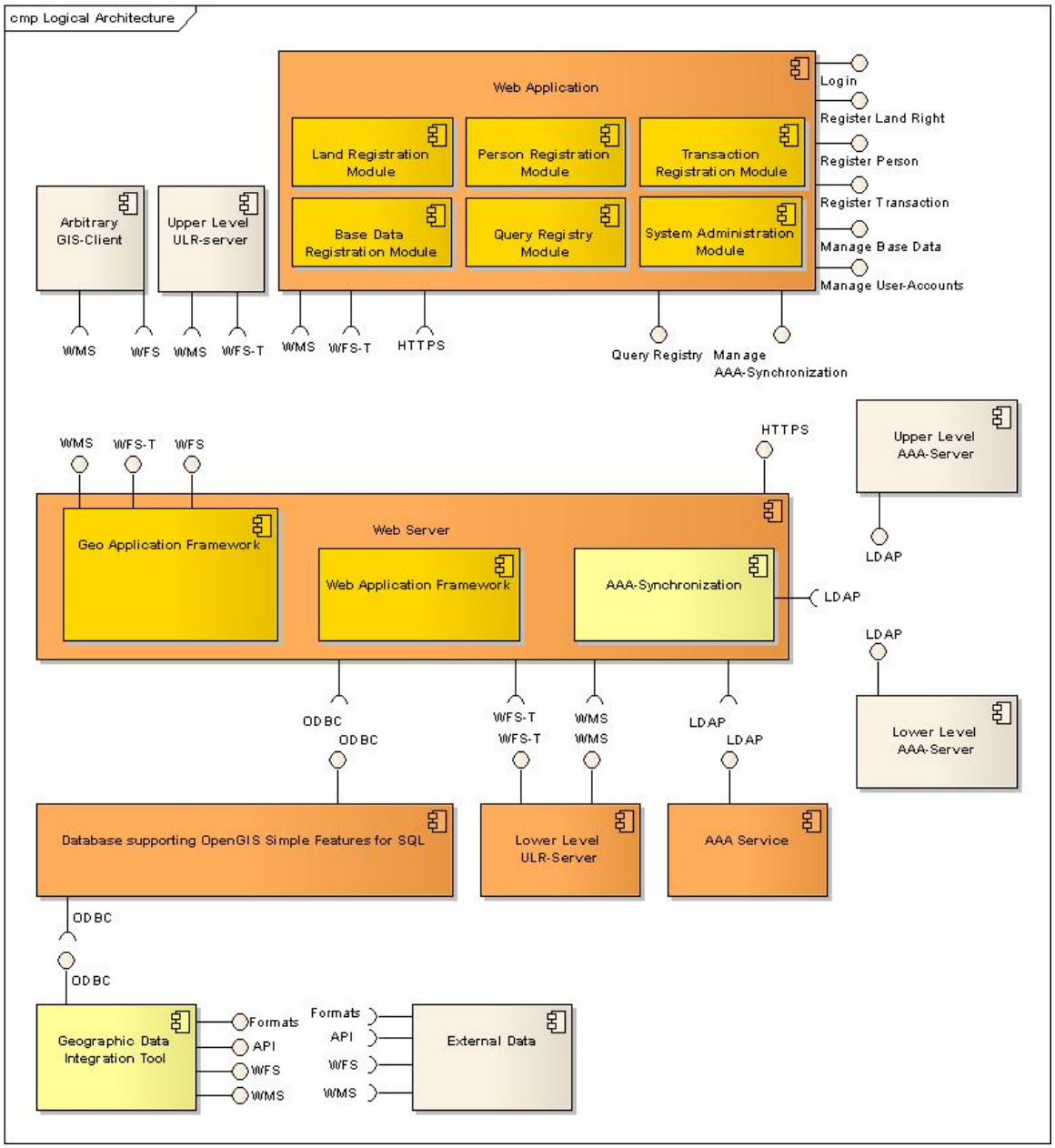


Abb. 5.1: Die logische Systemarchitektur der ULR-Applikation

WFS bzw. HTTP/HTTPS kommuniziert.

Für die gewählte 3-Schicht-Architektur sprechen folgende Gründe:

- Die einzelnen Software-Komponenten, die in den Schichten zum Einsatz kommen, können bei Bedarf ausgewechselt werden. Dies ist sinnvoll, da ein System sich immer weiterentwickeln wird und neue Anforderungen an Funktionalität entstehen. Auch wird es Komponenten geben, die die an sie gestellten Anforderungen besser und schneller erfüllen. Die 3-Schichten-Architektur erlaubt ein solches Auswechseln ohne Komplikationen.

- Durch die Trennung der Client-Schicht von den übrigen Schichten und durch die Verwendung von standardisierten Schnittstellen ist der Einsatz von beliebigen Clients möglich, ohne dass an diese weitergehende Anforderungen gestellt werden müssen. Auch ist es möglich, dass mehrere Clients gleichzeitig bedient werden können.

- Durch den gewählten Ansatz ist es insbesondere möglich, einen Client als Webapplikation in einem beliebigen modernen Webbrowser laufen zu lassen. Dies hat den grossen Vorteil, dass auf dem Client keine spezielle Software installiert werden muss (ein Webbrowser kann als gegeben vorausgesetzt werden.) und dass verschiedene Betriebssysteme eingesetzt werden können.

- Eine Drei-Schichten-Architektur lässt sich relativ einfach skalieren. So können alle drei Schichten auf dem gleichen physischen Rechner betrieben werden. Bei Bedarf können die Schichten aber auch verteilt auf verschiedenen Rechnern laufen.

6 Prototyp

Dieses Kapitel beschreibt die Umsetzung der in Kapitel 3 bis 5 beschriebenen ULR-Applikation in einem einfachen Prototypen. Dies geschieht einerseits, indem die gewählte physische Architektur beschrieben wird und andererseits, indem einige Informationen zur Applikation selber gemacht werden. Es kann allerdings in dieser schriftlichen Form kein vollständiger Eindruck von der Prototyp-Umsetzung entstehen.

6.1 Ziele

Das Ziel der Umsetzung in einem Prototypen liegt darin, die entworfene ULR-Applikation erstmals umzusetzen und ihre Tauglichkeit zu bewerten. Konkret sollen die Use Cases (Abschnitt 4) und die logische Systemarchitektur (5) auf ihre Praktikabilität geprüft werden. Die dabei gemachten Erfahrungen werden festgehalten (Abschnitt 6.5) und dienen als Referenzwerte für zukünftige Umsetzungen und allfällige Weiterentwicklungen des Applikationsentwurfes. Eine weitere Absicht ist, dass der Prototyp für das Gesamtprojekt ULR ein wichtiges Demonstrationsobjekt ist, mit dem interessierten Partnern auch plastisch gezeigt werden kann, um was es bei der ULR geht. Im Rahmen der verfügbaren Zeit konnte nicht der gesamte Applikationsentwurf umgesetzt werden. Es wurden aber die wichtigsten Bereiche berücksichtigt.

6.2 Physische Systemarchitektur

Ausgehend von der logischen Systemarchitektur (Abschnitt 5) wurde für die Umsetzung eine physische Systemarchitektur gewählt, die in Abbildung 6.1 dargestellt ist. Auf den ersten Blick wird klar, dass die Vorgaben der logischen Systemarchitektur grösstenteils erfüllt und dass für alle Komponenten Open Source-Produkte verwendet wurden.

6.2.1 Datenschicht

Als Datenbank dient die Kombination von PostgreSQL und PostGIS. Während PostgreSQL als Datenbank-Engine die eigentliche Datenverwaltung übernimmt, steht PostGIS für die räumlichen Fähigkeiten. PostGIS folgt dabei dem SFS-Standard, so dass bereits auf der Ebene der Datenhaltung die Einhaltung der geforderten Standards gewährleistet ist.

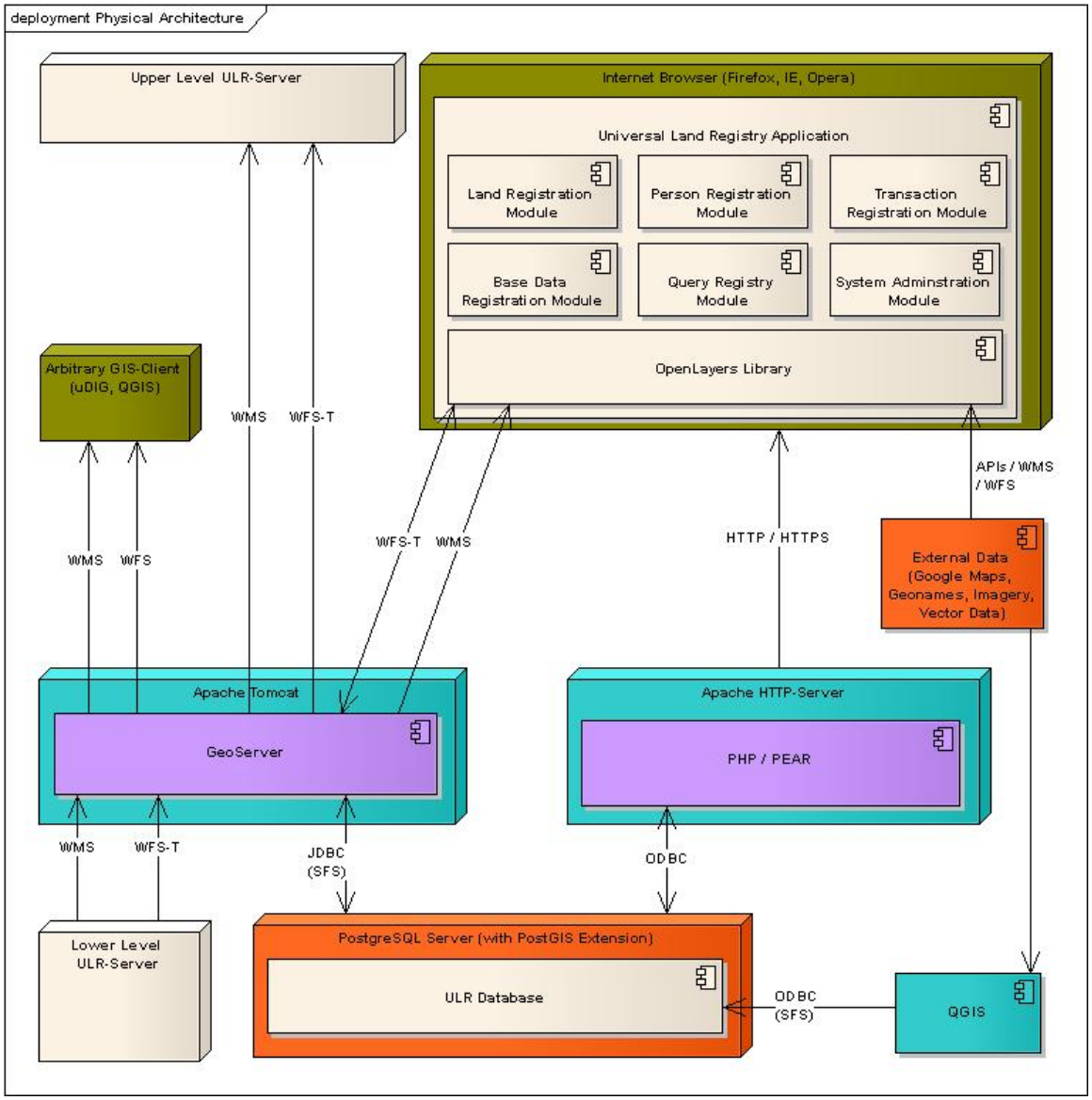


Abb. 6.1: Die Komponenten der physischen Systemarchitektur des Prototypen

Das vorgegebene ULR-Datenmodell (vgl. Abschnitt 3.4) konnte nicht komplett umgesetzt werden. Der Bereich der topologisch korrekten Parzellen wurde ausgeklammert, da die Topologie-Unterstützung von PostGIS noch in den Kinderschuhen steckt (REFRACTIONS RESEARCH 2005). Daher wurde nur die Verwaltung von Point-, MultiPoint-, Line-, MultiLine-, Polygon- und MultiPolygon-Geometrien umgesetzt. Ansonsten wurde das ULR-Datenmodell wie vorgegeben umgesetzt. Zusätzlich zur reinen Datenhaltung in der Datenbank wurde mit dem Einsatz von Views und Rules bereits in dieser Schicht Applikationslogik in die Datenbank eingebaut. Dies widerspricht der reinen Lehre einer 3-Schicht-Architektur, aber im Hinblick auf eine pragmatische und möglichst einfache Umsetzung ist dieses Vorgehen vertretbar.

Ein weiterer Punkt, der aus Zeitgründen nicht umgesetzt werden konnte, ist der AAA-Dienst. Geradezu prädestiniert für den Einsatz in diesem Bereich ist das Lightweight Directory Access Protocol (LDAP). Dabei werden die Daten der Benutzerverwaltung in einem sogenannten Verzeichnis (Directory) abgelegt und über das LDAP-Protokoll abgerufen. Eine kurze Recherche hat allerdings gezeigt, dass auf dem Open Source-Markt mit OpenLDAP und Apache Directory Server mindestens zwei Produkte existieren, die das LDAP-Protokoll unterstützen (OPENLDAP FOUNDATION 2008b, APACHE SOFTWARE FOUNDATION 2007a). Die Daten der Benutzerverwaltung des Prototypen wurden stattdessen in Tabellen in der Datenbank abgelegt. Da der AAA-Dienst nicht implementiert wurde, konnte natürlich auch die Synchronisierung des AAA-Dienstes mit den ULR-Knoten ober- oder unterhalb nicht realisiert werden. Auch hierzu gibt es in den erwähnten Produkten Funktionalitäten, mit denen die gestellten Anforderungen (vgl. Abschnitt 3.2) erfüllt werden können (OPENLDAP FOUNDATION 2008a, APACHE SOFTWARE FOUNDATION 2007b).

Die Synchronisation des lokalen ULR-Knotens mit den übrigen ULR-Knoten ist im Prototyp so umgesetzt, dass die Knoten miteinander über WFS kommunizieren. D.h. ein lokaler ULR-Knoten kann ganz einfach den WFS-Dienst eines untergeordneten Knoten einbinden. Allerdings nur wenn zwischen diesen Knoten eine Verbindung besteht. In einer kompletten Umsetzung müsste basierend auf den WFS-Diensten ein kompletter Synchronisationsdienst entwickelt werden, der den Datenabgleich steuert.

Die Integration von externen Geodaten wurde mit der Desktop GIS-Software QGIS realisiert, welche den direkten Datenimport in eine PostGIS-Datenbank unterstützt (QGIS COMMUNITY 2008). Damit ist es möglich, beliebige externe Geodaten aufzubereiten und sie in der Datenbank abzulegen.

6.2.2 Applikationsschicht

In der mittleren Schicht befinden sich zwei Hauptkomponenten: ein Apache HTTP-Server und ein Apache Tomcat-Server. Der HTTP-Server beinhaltet eine Instanz der PHP-Skriptsprache, mit der die eigentliche ULR-Webapplikation in Form von HTML-Seiten generiert wird. Die PHP-Erweiterung PEAR bindet eine Vielzahl an Bibliotheken ein, mit denen grosse Teile der benötigten Funktionalität abgedeckt werden können.

Parallel wird ein Java-basierter Apache Tomcat-Server eingesetzt, in dem das Geo-Applikationsframework Geoserver beheimatet ist. Geoserver ist in der Lage, räumliche

Daten via verschiedene Schnittstellen auszuliefern und damit Geowebdienste zu ermöglichen. Für diese Zwecke sehr wichtig sind die Optionen WMS und WFS-T. Damit werden auch auf dieser Ebene die geforderten OGC-Standards erfüllt. Die Kommunikation zwischen dieser Schicht und der Datenbank läuft über die Standard-Schnittstellen JDBC (Apache Tomcat) und ODBC (Apache HTTP-Server). Die Hauptaufgabe von Geoserver ist es, die ULR-Daten (gemäss Datenmodell) in Form von WFS- und WMS-Diensten zur Verfügung zu stellen. Im Bereich WFS ist explizit auch ein WFS-Transactional-Dienst gefordert, da über diesen Weg die Daten auch editiert werden sollen. Der WFS-Dienst wird in der Version 1.1 implementiert. Abbildung 6.2 zeigt beispielhaft den Featuretype vw_spatialunits_multipolygon wie er vom WFS-Dienst auf den Request describeFeatureType zurückgegeben wird. Parallel dazu werden die (räumlichen) Daten auch als WMS-Dienst in der Version 1.1.1 zur Verfügung gestellt. Geoserver bietet aber nicht nur die eigentlichen ULR-Daten per Webdienst zur weiteren Nutzung in der Webapplikation an sondern auch die in der Datenbank integrierten externen Geodaten. Dazu wurde allerdings auf die Entwicklung eines speziellen Konfigurationsinterfaces in der Webapplikation verzichtet. Dieser Zusatzaufwand wurde als zu gross eingeschätzt. Vielmehr muss der Data Administrator die Webdienste über die herkömmlichen Geoserver-Konfigurationsinterfaces bereitstellen, welche jedoch sehr komfortabel sind.

6.2.3 Präsentationsschicht

Auf der obersten Schicht ist die eigentliche ULR-Webapplikation angesiedelt. Sie besteht hauptsächlich aus durch den HTTP-Server ausgelieferten und von PHP erzeugten HTML-Seiten, die in einem beliebigen Browser angezeigt werden können. In den Teilen der Applikation, in denen räumliche Daten angezeigt oder editiert werden, kommt im Browser die Openlayers-Bibliothek ins Spiel. Openlayers ist eine Javascript-Bibliothek, die es auf relativ einfache Art und Weise ermöglicht, komplexe Kartenanwendungen zu entwickeln. Openlayers ist in der Lage, eine Vielzahl an Schnittstellen zu bedienen und in eine Webapplikation zu integrieren. Dazu gehören die OGC-Standards WMS und WFS, aber auch neuaufkommende Standards wie GeoRSS, GeoJSON und KML. Ausserdem können mit Openlayers auch Daten über die APIs kommerzieller Kartenanbieter wie Google Maps, Yahoo! Maps oder Windows Live Search Maps eingebunden werden (OPENLAYERS COMMUNITY 2008). Openlayers nimmt daher eine zentrale Stelle im Design des Webapplikation ein. Die Kommunikation zwischen der Webapplikation und der Applikationsschicht geschieht über WMS, WFS sowie HTTPS. Neben der Webapplikation ist es möglich, die angebotenen Dienste (WMS und WFS) auch mit beliebigen anderen Clients zu nutzen, die die Standards unterstützen. Zu Testzwecken wurden die Open Source-Applikationen uDIG und QGIS eingesetzt.

6.3 Sicherheit

Die in Kapitel 2.4.2.2 geforderten fünf Ebenen der Sicherheit konnten nicht vollständig implementiert werden. Die Transportsicherheit wurde mit dem Einsatz von HTTPS gewähr-

```
<xsd:schema elementFormDefault="qualified" targetNamespace="http://www.peterschaer.ch/ulr">
    <xsd:import namespace="http://www.opengis.net/gml" schemaLocation="http://localhost:8180/geoserver/wfs/schemas/gml/3.1.1/base/gml.xsd"/>
    <xsd:complexType name="vw_spatialunits_multipolygonsType">
        <xsd:complexContent>
            <xsd:extension base="gml:AbstractFeatureType">
                <xsd:sequence>
                    <xsd:element maxOccurs="1" minOccurs="0" name="spatialunit_multipolygon_id" nillable="true" type="xsd:int"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="geometry" nillable="true" type="gml:MultiSurfacePropertyType"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="spatialunit_id" nillable="true" type="xsd:int"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="accuracy" nillable="true" type="xsd:int"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="boundarytext" nillable="true" type="xsd:string"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="coord_dimension" nillable="true" type="xsd:int"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="date" nillable="true" type="xsd:date"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="estimatedsize" nillable="true" type="xsd:decimal"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="method" nillable="true" type="xsd:string"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="source" nillable="true" type="xsd:int"/>
                    <xsd:element maxOccurs="1" minOccurs="0" name="surveyor" nillable="true" type="xsd:string"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
    <xsd:element name="vw_spatialunits_multipolygons" substitutionGroup="gml:_Feature" type="ulr:vw_spatialunits_multipolygonsType"/>
</xsd:schema>
```

Abb. 6.2: Der GML-Featuretype vw_spatialunits_multipolygons

leistet. Die eingesetzten Apache-Server unterstützen dieses Protokoll schon seit langem. Auf die Implementierung der Nachrichtensicherheit (z.B. mit XML Encryption) wurde aus Zeitgründen verzichtet. Die Bereiche Authentifizierung und Autorisierung wurden wie bereits erwähnt nicht mit dem eigentlich geforderten AAA-Dienst und über das LDAP-Protokoll umgesetzt. Stattdessen wurde eine einfache Benutzerverwaltung aufgebaut, die auf Tabellen in der Datenbank basiert. Sowohl die Authentifizierung als auch die Autorisierung laufen über diese Benutzerverwaltung. Geoserver unterstützt noch keinen der in Kapitel 2.4.2.2 erwähnten Standards (z.B. WAS, WSS etc.), so dass hier auf die eingebaute Benutzerverwaltung zurückgegriffen werden musste (GEOSERVER COMMUNITY 2008). Durch den Wegfall des AAA-Dienstes konnte die fünfte Sicherheitsebene – das Accounting – nicht wie gefordert umgesetzt werden. Hier bot sich jedoch die Möglichkeit, dieses zumindest teilweise auf Datenbankebene mit Triggern nachzubilden.

6.4 Prototyp-Applikation

Es ist aus Platzgründen nicht sinnvoll, die gesamte Webapplikation in aller Ausführlichkeit zu beschreiben. Deshalb werden anhand von Screenshots einige wenige Ausschnitte aus der Webapplikation gezeigt.

Die Webapplikation ist sehr einfach aufgebaut. Für jede Klasse des Datenmodells gibt es eine Seite, mit der eine neue Instanz dieser Klasse registriert werden kann. Als Beispiel hierfür sei Abbildung 6.3 erwähnt, die das Formular zeigt, mit dem eine natürliche Person registriert werden kann. Die anderen Formulare für juristische Personen, Landrechte und Parzellen sehen analog aus. Für die Anzeige der Parzellen ist natürlich ein Formular oder eine Liste nicht geeignet. Hier muss eine Kartendarstellung zum Zug kommen. In der Webapplikation wurden alle Kartenanwendungen mit der Openlayers-Bibliothek realisiert. Abbildung 6.4 zeigt eine solche Kartenanwendung, in der vor dem Hintergrund eines Google Maps-Luftbildes einige Parzellen aus der Datenbank in gelb angezeigt werden. Das grosse Polygon rechts unten ist selektiert. Seine Eigenschaften werden in der rechten Bildschirmhälfte angezeigt. Für das selektierte Polygon werden alle erfassten Vertices als gelbe Kreise dargestellt. Diese Vertices können nun mit der Maus angeklickt und beliebig verschoben oder gelöscht werden. Es können auch neue Vertices eingefügt werden. Damit kann die Geometrie einer Parzelle sehr genau digitalisiert werden. Auf eine Snapping-Funktion musste verzichtet werden, da die Openlayers-Bibliothek diese Funktion in der verwendeten Version (2.5) noch nicht integriert hat. In einer der nächsten Versionen soll dies allerdings nachgeholt werden, so dass ein weiterentwickelter Prototyp diese Anforderung auch erfüllen wird. Rechts oben in der Karte ist der sogenannte Layermanager sichtbar, mit dem die verfügbaren Ebenen ein- und ausgeschaltet werden können. Im gezeigten Ausschnitt sind nur wenige Layer verfügbar, es können aber problemlos zusätzliche Layer hinzugefügt werden.

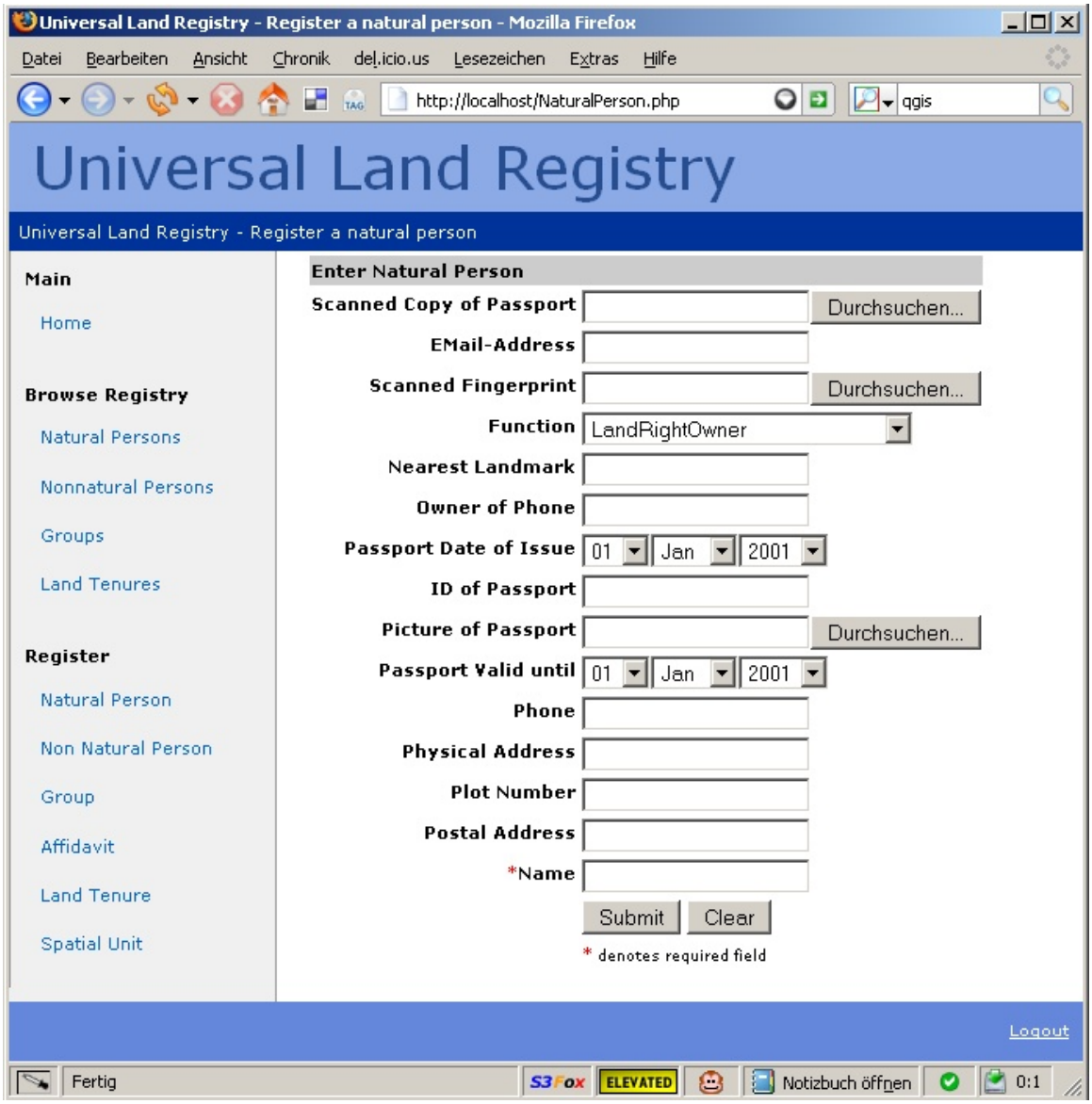


Abb. 6.3: Formular zur Registrierung einer natürlichen Person

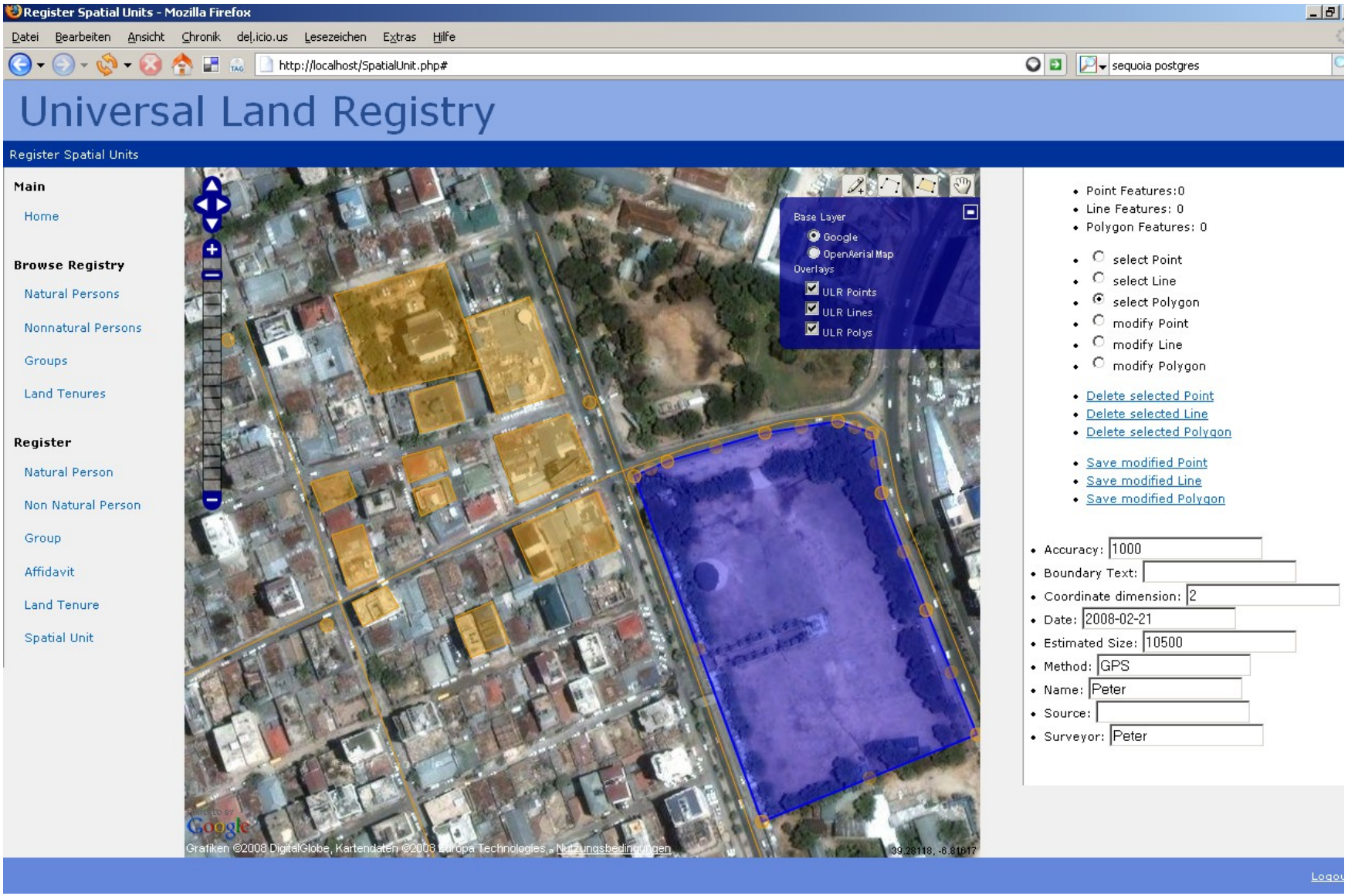


Abb. 6.4: Kartenanwendung zur Editierung einer Parzelle

6.5 Fazit

Es hat sich gezeigt, dass die Implementierung eines Prototypen, der alle Use Cases und Anforderungen erfüllen soll, zeitlich deutlich aufwendiger ist, als zu Beginn geschätzt wurde. Aus diesem Grund musste auf einige Punkte verzichtet werden. Ausserdem haben technische Limitationen zum Verzicht auf andere Bereiche geführt. Letztendlich konnte also nur ein Teil der Anforderungen und Use Cases umgesetzt werden. Nichtsdestotrotz ergeben sich aus dem vorliegenden Prototypen einige wertvolle Hinweise für die Weiterentwicklung des Prototypen im Speziellen und der ULR-Applikation im Allgemeinen.

Grundsätzlich kann gesagt werden, dass die logische Systemarchitektur so defniert ist, dass sie mit existierenden Open Source-Komponenten umgesetzt werden kann. Es sind keine Anpassungen notwendig. Es wurde auch gezeigt, dass der Anwender praktisch die gesamte Kommunikation mit der ULR über den Web-Browser erledigen kann. Es ist keine zusätzliche Software notwendig. Es kann ebenfalls festgehalten werden, dass die beschriebenen Use Cases umgesetzt werden konnten. Es sind keine grösseren Ergänzungen notwendig. Es ist aber denkbar, dass die Use Cases noch präziser und detailierter gestaltet werden, damit sie noch besser als Implementierungsanleitung dienen können.

Die 3-Schichten-Systemarchitektur der Applikation führt dazu, dass eine grosse Zahl an verschiedenen Komponenten zum Einsatz kommen. Für die Umsetzung muss entsprechendes Fachwissen zu all diesen Komponenten vorhanden sein oder geschaffen werden. Die Anzahl der Komponenten kann im Prototypen sicherlich noch reduziert werden. Der Einsatz von zwei parallelen Servern in der Applikationsschicht (Apache HTTP-Server und Apache Tomcat) ist keinesfalls ideal und wurde auch nur deswegen gewählt, weil im Bereich PHP mehr Fachwissen zur Verfügung stand als im Bereich Java. Für weitere Umsetzungen wird es sicher sinnvoll sein, die Webapplikation in einem der zahlreichen Java-basierten Frameworks (Struts, Cocoon etc.) zu realisieren. Auf Apache HTTP-Server und PHP/PEAR kann in diesem Falle dann verzichtet werden.

Der Einbezug von externen Basisdaten sei es über Webdienste (WMS, WFS etc.) oder über APIs (Google Maps, Geonames etc.) ist mit der Openlayers-Bibliothek ohne Probleme und mit so geringem Aufwand möglich, dass es Sinn macht, solche Daten in existierenden Applikationen und Datenbanken wie Openstreetmap, Openaerialmap oder Geonames zu verwalten und per API oder Webdienst einzubinden. Dies hat den Vorteil, dass auf die Interfaces dieser externen Datenbanken zurückgegriffen werden kann und für die ULR-Applikation keine Eigenentwicklung notwendig ist.

Um den Implementierungsaufwand zu reduzieren, ist es generell sinnvoll, so oft wie nur möglich auf bestehende Komponenten und Bibliotheken zurückzugreifen. Das Programmieren von Eigenentwicklungen ist aufwendig und sollte nur dort angewendet werden, wo es unbedingt nötig ist. Der Aufwand für die Abstimmung und Koordination der verschiedenen Komponenten ist nicht zu vernachlässigen, ist aber deutlich geringer als derjenige für Eigenentwicklungen. Deshalb kann für zukünftige Umsetzungen nur empfohlen werden, gerade für die Komponente des Web Application Frameworks ein Produkt zu wählen, dass dem Entwickler möglichst viel Programmierarbeit abnimmt. Die Kombination aus PHP und PEAR ist nicht die effizienteste und hat bei der Entwicklung des Prototypen einigen Aufwand für eigene Programmierung nötig gemacht. Interessante Produkte in diesem

Bereich sind sicherlich Ruby on Rails oder Grails.

7 Schlussfolgerungen

In diesem Kapitel werden die in der Einführung formulierten Hypothesen überprüft, das gewählte Vorgehen beleuchtet sowie ein Blick in die Zukunft geworfen.

7.1 Beurteilung der Hypothesen

Hypothese 1: Die Universal Land Registry – wie von MITHÖFER (2008) beschrieben – ist mit existierenden IT- und GIS-Mitteln praktisch umsetzbar.

Die Kapitel 3 bis 4 haben gezeigt, dass die Vorgaben der ULR (MITHÖFER 2008) durchaus umgesetzt werden können. Zwar sind diese Vorgaben anspruchsvoll, doch die gewählte 3-Schichten-Architektur erlaubt den Einsatz verschiedenster Komponenten, von welchen der GIS- und IT-Markt mittlerweile eine Fülle anbietet.

Die Beschreibung von verschiedenen Sichten auf das System mit UML-Diagrammen ist ein sehr gutes Instrument, um die Umsetzung in all ihren Aspekten definieren zu können. Diese Systembeschreibung ist dann sehr hilfreich, wenn es darum geht, die richtigen Komponenten für eine Umsetzung auszuwählen und wenn die Systembeschreibung mit anderen Personen diskutiert werden soll.

Die grosse Zahl an Komponenten, die in einem solchen System eingesetzt werden, führt zu einem hohen Überwachungs- und Koordinationsaufwand. Die Interoperabilität zwischen den Komponenten ist durch den Einsatz von offenen Standards gewährleistet und es können einzelne Komponenten auch problemlos gegen funktional gleichwertige andere Produkte ausgetauscht werden. Zur Überwachung des Zusammenspiels der Komponenten könnten vielleicht zusätzliche Dienste eingesetzt werden. Dies ist jedoch nicht Teil des aktuellen Designs.

Aus dem aktuellen Design bewusst ausgeklammert wurden Aspekte, die mit der Skalierbarkeit des Systems zusammenhängen. Dies aus dem Grunde, weil eine Test-Umgebung in einem grösseren Rahmen nicht vorlag. Das vorliegende Design kann nicht garantieren, dass eine Umsetzung auf nationaler Ebene tatsächlich performant geschehen kann. Doch der Einsatz von (auch in grossen Projekten) bewährten Komponenten erhöht die Wahrscheinlichkeit erheblich, dass auch ein Betrieb auf nationaler Ebene performant sein wird. Gerade in der nun folgenden Pilotphase des ULR-Projektes müssen solche Fragestellungen ganz intensiv betracht und getestet werden.

Hypothese 2: Der Einsatz von Open Source-Komponenten erlaubt eine Um-

setzung der ULR, die den relevanten IT- und GIS-Standards folgt.

Wie in Hypothese 1 erwähnt, bietet der Markt genügend Produkte an, mit denen das technische Design der ULR umgesetzt werden kann. Dies trifft nicht nur auf den Gesamtmarkt zu sondern auch auf den Open Source-Markt. Die in der prototypischen Umsetzung gewählten Komponenten stammen allesamt aus der Open Source-Welt. Sowohl im GIS-Bereich als auch im traditionellen IT-Bereich ist die Zahl der Open Source-Komponenten so gross, dass meistens eine Auswahl getroffen werden muss. Es ist auch damit zu rechnen, dass sich die Open Source-Komponenten in Zukunft sowohl in Sachen Qualität als auch Quantität noch weiter verbessern werden. Damit werden in der Pilotphase des ULR-Projekts bereits wieder andere oder aktualisierte Komponenten eingesetzt werden. Dies zeigt einmal mehr, dass der gewählte modulare, verteilte Ansatz für die Umsetzung der richtige ist. Nur wenn einzelne Komponenten ausgetauscht werden können, wird die ULR auch in Zukunft mit wichtigen Trends mithalten und sich weiterentwickeln.

Hypothese 3: Die Abstützung auf IT- und GIS-Standards garantiert, dass die ULR langfristig eingesetzt werden und als Kern einer künftigen nationalen Geodateninfrastruktur Tansanias dienen kann.

Diese Hypothese kann bestätigt werden. Es wurde gezeigt, dass ein modulares, verteiltes Design, wie es hier für die ULR entworfen wurde, nur mit offenen Standards funktionieren kann. Der Einsatz von proprietären (nur für die ULR entwickelten) Standards macht keinen Sinn. Zwar könnte auf diese Weise zum jetzigen Zeitpunkt eine ULR-Umsetzung entworfen werden. Diese könnte jedoch nur mit grossem Aufwand an neue technologische Trends angepasst werden. Eine Nachhaltigkeit wäre nicht gegeben. Aus diesem Grund wurde beim Entwurf des technischen Designs grossen Wert auf Standards gesetzt.

Da es momentan wenig bis gar keine Arbeiten an einer nationalen Geodateninfrastruktur Tansanias gibt, kann der zweite Teil der Hypothese nicht direkt beantwortet werden. Es darf aber berechtigterweise angenommen werden, dass sich eine tansanische Geodateninfrastruktur wie andere Geodateninfrastrukturen auch auf offene Standards wie diejenigen des Open Geospatial Consortium stützen wird. Deshalb spricht einiges dafür, dass auch dieser Teil der Hypothese bestätigt werden kann.

7.2 Synthese

Dieser Arbeit lag das Hauptziel zugrunde, ein technisches Design für die Umsetzung der ULR zu entwerfen. Die einzelnen Schritte der gewählten Vorgehensweise können mit folgenden Punkten zusammengefasst und bewertet werden:

- Literaturrecherche zur Erarbeitung der notwendigen Grundlagen und Konzepte. Dies war ein wichtiger Schritt, um einschätzen zu können, in welche Richtung eine technische Umsetzung gehen kann. Sehr interessant war auch die Tatsache, dass gerade im Bereich Landregistrierung einzelne Arbeiten (z.B. RUTAMO 2006 oder BRENT-JENS 2004) existieren, die in einem ganz ähnlichen Themenfeld wie die vorliegende

Arbeit aktiv waren. Der Fokus auf ein nationales Framework mit verteilter Architektur, wie es die ULR ist, ist jedoch neu.

- Anforderungsanalyse. Zusammen mit den Anwendungsfällen und der logischen Systemarchitektur ist dies der Kern der vorliegenden Arbeit. Nur das Studium aller relevanten Vorgaben in der Literaturrecherche und deren Ausformulierung als konkrete Anforderungen an das System ermöglicht die Abdeckung aller Bedürfnisse. Denn letztlich bestimmen nicht die Technologien das Aussehen des Systems sondern die Bedürfnisse.

- Identifizierung aller Anwendungsfälle. Die Beschreibung des Systems aus Benutzersicht war sehr wichtig aber auch nicht einfach, da der Benutzer hypothetischer Natur war. Weder gibt es in Tansania bereits ein funktionierendes System zur Landregistrierung, dessen Anwender hätten befragt werden können, noch hätte die grosse räumliche Distanz zu Tansania dies möglich gemacht. Eine Befragung von potenziellen Anwendern hätte den Prozess der Identifizierung der Anwendungsfälle mit Sicherheit bereichert und den Anwendungsfällen zu höherer Präzision und mehr Detail verholfen.

- Entwurf der logischen Systemarchitektur. In diesem Entwurf wurden die Anforderungen und die Anwendungsfälle in einem logischen Design zusammengefasst. Hier gibt es natürlich mehrere Lösungen. Um die hier vorliegende Lösung zu entwerfen, wurde parallel dazu der Prototyp entwickelt, um gewisse Probleme früh zu erkennen und in den Entwurf zurückfliessen zu lassen. Dieses iterative Vorgehen hat sich sehr bewährt.

- Prototyp-Umsetzung. Die praktische Umsetzung in einem Prototypen war als hilfreiche Ergänzung gedacht. Aus zeitlichen Gründen konnte nicht für alle Teile des Systems eine praktische Umsetzung erreicht werden. Doch in den Bereichen, die umgesetzt wurden, kann festgestellt werden, dass das Design durchaus praxistauglich ist.

In praktisch jedem Schritt wurden ein oder mehrere UML-Diagrammtypen eingesetzt. Die grosse Zahl an möglichen UML-Diagrammen ermöglichte es, genau die Diagramme auszuwählen, die es erlauben, das System möglichst präzise zu entwerfen. Der Einsatz einer formalisierten, visuellen Sprache wie UML ist daher sehr fruchtbar. Es kann damit auch erwartet werden, dass Personen, die sich in Zukunft mit der technischen Umsetzung der ULR befassen werden, ohne Probleme die Ergebnisse der vorliegenden Arbeit verstehen, anwenden und weiterentwickeln können.

7.3 Ausblick

Mit der vorliegenden Arbeit sowie der Arbeit von MITHÖFER (2008) liegt nun eine umfassende Beschreibung des ULR-Frameworks auf formaler, organisatorischer und technischer Ebene vor. Die Ergebnisse wurden auch bereits einem Fachpublikum vorgestellt. Nun gilt

es, das ULR-Projekt weiterzutreiben. Gemäss dem Projektpapier von HUBER (2006a) wird das Projekt in eine nächste Phase eintreten. Das Ziel wird dabei sein, die vorliegenden Arbeiten bei den tansanischen Entscheidungsträgern im MKURABITA-Programm vorzustellen. Gleichzeitig muss die Suche nach Geldgebern für das Projekt intensiviert werden. Schliesslich soll diese nächste Projektphase mit einem Pilotprojekt abgeschlossen werden. Dabei wird die vorliegende Arbeit sicherlich intensiv auf ihre Tauglichkeit geprüft werden.

Auf technologischer Ebene ist für die nächste Zukunft zu erwarten, dass im Bereich der Interoperabilität gerade die Open Source-Komponenten einen wichtigen Schritt vorwärts machen werden. Die erwähnten Standards werden noch besser unterstützt werden. Gleichzeitig muss bei der Weiterentwicklung der ULR und des technischen Designs auch ein Auge auf neu auftretende Standards geworfen werden (z.B. KML, GeoRSS, GeoJSON, Security-Standards etc.). Solche Standards befinden sich noch in der Entwicklungsphase, können aber auch für die ULR sehr fruchtbar sein. Es ist jedoch zum jetzigen Zeitpunkt nicht möglich abzuschätzen, welche dieser Standards sich durchsetzen werden.

8 Literaturverzeichnis

ABDULAI, Raymond Talinbe, 2006: Is land title registration the answer to insecure and uncertain property rights in sub-Saharan Africa? RICS Research paper series, Volume 6, Number 6. Wolverhampton.

APACHE SOFTWARE FOUNDATION, 2007a: ApacheDS v1.0 Features. http://directory.apache.org/apacheds/1.0/apacheds-v10-features.html (21.04.2008).

APACHE SOFTWARE FOUNDATION, 2007b: ApacheDS v1.5 – Mitosis. http://directory.apache.org/apacheds/1.5/mitosis.html (21.04.2008).

ASHENFELTER, John Paul, 2002: Database Access Protocols. http://www.ddj.com/184412642 (21.04.2008).

BRENTJENS, Thijs, 2004: OpenGIS Web Feature Services for editing cadastral data. Analysis and practical experiences. Delft University of Technology, Delft.

BUEHLER, K.; MCKEE, L, 1998: The OpenGIS Guide: Introduction to Interoperable Geoprocessing and the OpenGIS Specification.

CHUNITHIPAISAN, Sanphet; JAMES, Philip; PARKER, David; MAJEED, Zainai Abdul; ABELE, Simon, 2003: Geospatial Interoperability via the Web: Supporting Land Administration in Kuala Lumpur. Map Asia Conference 2003. http://www.gisdevelopment.net/application/lis/overview/pdf/ma0353.pdf

CLARKE, George; WAITE, Mark; BOYD, Graham; KALLONGA, Emmanuel; LINGSON, Adam; CONNELLY, Philip; MPANGALA, Nathan; MAHON, Elaine, 2007: Poor People's Wealth. Plain language information about Tanzania's Property and Business Formalisation Programme MKURABITA. groups.google.com/group/mkurabita_debate/web/MKURABITA-UnofficialGuide-Hakikazi-Feb2007-English.pdf (21.04. 2008).

DE SOTO, Hernando, 2006: The Challenge of Connecting Informal and Formal Property Systems. Some reflections based on the case of Tanzania. In: CHENEVAL, Francis; DE SOTO, Hernando (Hrsg.), 2006: Realizing Property Rights. Swiss Human Rights Book, Vol. I. Zürich, S. 18-67.

EBERLE, Hanspeter, 2006: Realisierung eines Umweltinformationssystem mit IT-Standard Methoden. Master Thesis, Universitätslehrgang "Geographical Information Science

& Systems“ (UNIGIS MSc), Zentrum für GeoInformatik (Z_GIS), Paris Lodron-Universität Salzburg.

FIELDING, Roy et al., 1999: Hypertext Transfer Protocol – HTTP/1.1. ftp://ftp.isi.edu/in-notes/rfc2616.txt (21.04.2008).

GEOSERVER COMMUNITY, 2008: Security subsystem. http://geoserver.org/display/GEOSDOC/5+Security+subsystem (21.04.2008).

HUBER, Martin, 2006a: National Spatial Data Infrastructure of Tanzania to Support a Transparent Land Registration Process. Unveröffentlichtes Projektpapier.

HUBER, Martin, 2006b: UNIGIS Modul “OpenGIS und verteilte Geoinformationsverarbeitung“ – Lektion 6: Weitere OGC-Dienste. Unveröffentlichte Vorlesungsunterlagen. Salzburg.

HUBER, Martin, 2006c: UNIGIS Modul “OpenGIS und verteilte Geoinformationsverarbeitung“ – Lektion 4: Simple Features. Unveröffentlichte Vorlesungsunterlagen. Salzburg.

HUBER, Martin; MITHÖFER, Klaus; SCHÄR Peter; HARVEY Francis; MUKASA Oscar, 2008: Universal Land Registry to Support Independent Economic Development in Tanzania. GSDI-10: Tenth International Conference for Spatial Data Infrastructure. St. Augustine, Trinidad, 25.-29. Februar 2008. http://www.gsdi.org/gsdi10/papers/TS17.1paper.pdf. (21.04.2008).

JACKSON, Jonathan, 2002: Technology as a problem in Southern African land tenure reform. Coming to terms with De Soto's criticism of technicians. FIG Symposium on Land Redistribution in Southern Africa. Pretoria, 6.-7. November 2002.

JOHANSSON, Johan, 2005: Improving Access to Geographic Information. Exploring the National Spatial Data Infrastructure Initiative in Tanzania. Thesis for the C-Seminar in Peace and Conflict Studies. University of Umea. Umea.

JOHNER, Heinz; BROWN, Larry; HINNER, Franz-Stefan; REIS, Wolfgang; WESTMAN, Johan, 1998: Understanding LDAP.

KAUFMANN, Jürg; STEUDLER, Daniel, 1998: CADASTRE 2014. A Vision for a Future Cadastral System. FIG. http://www.fig.net/cadastre2014/translation/c2014-english.pdf. (21.04.2008).

LEMMEN, Christiaan; VAN OOSTEROM, Peter, 2006: Version 1.0 of the FIG Core Cadastral Domain Model. Shaping the Change. XXIII FIG Congress. München, 8.-13. März 2006. http://www.fig.net/pub/fig2006/papers/ts12/ts12_02_lemmen_vanoosterom_0605.pdf (21.04.2008).

LEMMEN, Christiaan; AUGUSTINUS Clarissa; VAN OOSTEROM, Peter; VAN DER MOLEN, Paul, 2007: The Social Tenure Domain Model – Design of a First Draft Model. XXX FIG General Assembly and Working Week. Hongkong, 13.-17. Mai 2007. http://www.fig.net/pub/fig2007/papers/ts_1a/TS01A_01_lemmen_augustinus_oosterom_molen_1373.pdf (21.04.2008).

MITHÖFER, Klaus, 2008: Development of a GIS based property information System for Tanzania. Master Thesis, Universitätslehrgang "Geographical Information Science & Systems" (UNIGIS MSc), Zentrum für GeoInformatik (Z_GIS), Paris Lodron-Universität Salzburg. (in Bearbeitung).

MÜHLEMANN, Rolf, 2007: Konzeption eines Sicherheitsframeworks für eine Open Source-basierte Geodateninfrastruktur. Master Thesis, Universitätslehrgang "Geographical Information Science & Systems" (UNIGIS MSc), Zentrum für GeoInformatik (Z_GIS), Paris Lodron-Universität Salzburg.

MUTAMBO, Levi Shimi, 2003: The Unified Modelling Language (UML) in Cadastral System Development. International Institute for Geo-Information Science and Earth Observation, Enschede.

NPA (Norvegian People's Aid, Autor M. Waite), 2006: Property and Business Formalization Program – Reform Design Stage, Progress Report – September 2006. groups.google.com/group/mkurabita_debate/web/MATRIX%20OVERVIEW1106F.doc (21.04.2008).

OGC (Open Geospatial Consortium), 2004: OpenGIS Geography Markup Language (GML) Implementation Specification, Version 3.1.1
http://www.opengeospatial.org/standards/gml (21.04.2008).

OGC (Open Geospatial Consortium), 2005a: Web Feature Service Implementation Specification, Version: 1.1.0
http://www.opengeospatial.org/standards/wfs (21.04.2008).

OGC (Open Geospatial Consortium), 2005b: OpenGIS Filter Encoding Implementation Specification, Version 1.1.0
http://www.opengeospatial.org/standards/filter (21.04.2008).

OGC (Open Geospatial Consortium), 2006a: OpenGIS Web Map Server Implementation Specification, Version 1.3.0
http://www.opengeospatial.org/standards/wms (21.04.2008).

OGC (Open Geospatial Consortium), 2006b: OpenGIS Implementation Specification for Geographic information – Simple feature access – Part 2: SQL option, Version: 1.2.0
http://www.opengeospatial.org/standards/sfs (21.04.2008).

OMG (Object Management Group), 2005: Introduction to OMG's Unified Modeling Language (UML).
http://www.omg.org/gettingstarted/what_is_uml.htm (21.04.2008).

OPENLAYERS COMMUNITY, 2008: OpenLayers Examples.
http://openlayers.org/dev/doc/examples.html (21.04.2008).

OPENLDAP FOUNDATION, 2008a: Does slapd(8) support multi-master replication?
http://www.openldap.org/faq/data/cache/1240.html (21.04.2008).

OPENLDAP FOUNDATION, 2008b: What versions of the LDAP does OpenLDAP Software implement?
http://www.openldap.org/faq/data/cache/162.html (21.04.2008).

PILONE, Dan; PITMAN, Neil, 2005: UML 2.0 in an Nutshell.

QGIS COMMUNITY 2008: Features.
http://qgis.org/content/view/145/113/ (21.04.2008).

RAMSEY, Paul, 2007: The State of Open Source GIS.
http://refractions.net/expertise/whitepapers/opensourcesurvey (21.04.2008).

RÄTZMANN, Manfred, 1998: Three-Tier-Development.
http://www.dfpug.de/konf%5Ckonf_1998%5C09_tier%5Cd_tier/d_tier.htm (21. 04. 2008).

REFRACTIONS RESEARCH, 2005: Postgis Topology.
http://postgis.refractions.net/support/wiki/index.php?PostgisTopology (21. 04. 2008).

RUTAMU, Déo R., 2006: Implementation of the Core Cadastral Domain Model in a Distributed Environment using OpenGIS Standards. International Institute for Geo-information Science and Earth Observation, Enschede.

SCHÄLING, Boris, 2005: Der moderne Softwareentwicklungsprozess mit UML.
http://www.highscore.de/uml (21.04.2008).

SCHÜTZE, Emanuel, 2007: Stand der Technik und Potenziale von Smart Map Browsing im Webbrowser. Diplomarbeit, Studiengang Medieninformatik, Hochschule Bremen.

VEREINTE NATIONEN, 1948: Die Allgemeine Erklärung der Menschenrechte.
http://www.unhchr.ch/udhr/lang/ger.htm (21.04.2008).

WIKIPEDIA, 2008: WIKIPEDIA – The free Encyclopedia.
http://www.wikipedia.org (21.04.2008).

WORBOYS, M.F., 1995: GIS: a computing perspective. London.