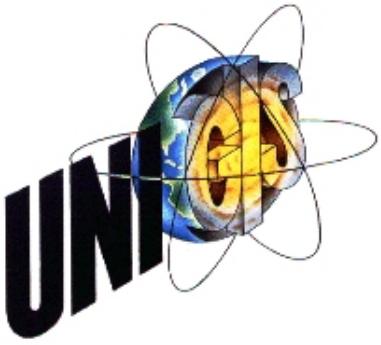


Master Thesis

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

zum Thema

„Konzeption und Entwicklung eines LPIS Web-GIS Prototypen für die Verwaltung landwirtschaftlicher Flächen auf Basis des ArcGIS Servers“

unter besonderer Berücksichtigung des topologischen Editierens

vorgelegt von

Dipl. Ing. Michael Bauer
U1331, UNIGIS MSc Jahrgang 2007

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

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

Ingolstadt, 29.10.2009

DANKSAGUNG

Ich möchte an dieser Stelle an allen Personen bedanken, die zur Verwirklichung dieser Master Thesis beigetragen haben.

Ein besonderer Dank gilt Herrn Dr. Peter Volk und Herrn Matthias Schulz der Firma GAF AG in München, durch die diese Arbeit erst entstehen konnte.

Danken möchte ich auch dem UNIGIS Team, dem Lehrgangsbüro und besonders Herrn Prof. Dr. Strobel für die Betreuung der Arbeit.

Ein spezieller Dank geht an meine Familie und Freunde, die mich in der Zeit des UNIGIS Studiums begleitet haben.

ERKLÄRUNG

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 entsprechend gekennzeichnet.

Ingolstadt, 28.10.2009

…….

Michael Bauer

KURZFASSUNG

Für die Beantragung von Subventionen für landwirtschaftlich genutzte Flächen hat sich in den Behörden der einzelnen Bundesländer die Verwendung von geographischen Informationssystemen etabliert. Die dafür von der GAF AG breitgestellte Lösung LaFIS-LFK ist eine Desktop Anwendung, die in Form einer browserbasierten Web-GIS Anwendung neu aufgesetzt werden soll.

Bei der Erfassung der landwirtschaftlichen Referenzflächen ist die Berücksichtigung von topologischen Beziehungen erforderlich. Diese Arbeit untersucht Möglichkeiten, wie die landwirtschaftlich genutzten Flächen mit einer ArcGIS Server Web-GIS Anwendung topologisch richtig erfasst werden können. Die Arbeit gibt einen Überblick über die ArcGIS Server Technologie, sowie über dessen Werkzeuge, Methoden und Bibliotheken.

In der Arbeit wird dargestellt, wie mit diesen Mitteln eine Web-GIS Anwendung mit Editierfunktionalitäten erstellt werden kann. Zudem zeigt sie die notwendigen Schritte zur Weiterentwicklung der Web-GIS Anwendung in einer Entwicklungsumgebung auf. Die Implementierung von topologischen Prüfmethoden ist anhand von Ablaufdiagrammen und Codebeispielen dargestellt. Ein Bewertung der erzielten Ergebnisse, sowie eine kritische Betrachtung bilden den Abschluss dieser Arbeit.

ABSTRACT

To apply for subsidies for agricultural land parcels the use of geographical information systems has been established in the authorities of the federal states. For this purpose the company GAF AG developed the desktop solution LaFIS-LFK, which should be re-drafted in the form of a browser-based Web-GIS application.

But the capturing of agricultural land parcels forces the consideration of topological relations. This thesis explores ways how farmland parcels can be captured in a topological correct way using an ArcGIS Server Web-GIS application. The thesis gives an overview of the ArcGIS Server technology, as well of its tools, methods, and libraries.

The work shows how a Web-GIS application with editing functionality can be created with the above mentioned technology. The thesis also outlines the necessary steps for the further development of the created Web-GIS application in a development environment. The implementation of topological methods is illustrated using flow charts and code examples. An evaluation of the results and a critical consideration will complete this work.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

LISTINGS (QUELLCODEBEISPIELE)

ABKÜRZUNGSVERZEICHNIS

A	
ADF	Application Developer Framework
AGS	ArcGIS Server
AJAX	Asynchronous JavaScript and XML
ALK	Amtliches Liegenschaftskataster
API	Application Programming Interface (Programmierschnittstelle)
C	
CGI	Common Gateway Interface
CSS	Cascading Stylesheets
D	
DB	Datenbank
DHTML	Dynamic HTML
DOM	Document Object Model
DOP	Digitales Orthophoto
DTD	Document Type Definition
E	
ESRI	Environmental Systems Research Institute
EU	Europäische Union
EWG	Europäische Wirtschaftsgemeinschaft
F	
FTP	File Transfer Protocol
G	
GAP	Gemeinsame Agrarpolitik (der Europäischen Union)
GIS	Geographisches Informationssystem
GML	Geographic Markup Language
GPS	Global Positioning System
H	
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I	
IACS	Integrated Administration and Control System (= InVeKoS)

IDE	Integrated Development Envirionment (dt.: integrierte Entwicklungsumgebung)
InVeKoS	Integriertes Verwaltungs- und Kontrollsystem (= IACS)
ITK	Informations- und Telekommunikationstechnologie
J	
JRE	Java Runtime Environment
JS	JavaScript
JSF	JavaServer Faces
JSP	JavaServer Pages
JVM	Java Virtual Machine
L	
LaFIS	Landwirtschaftliches Flächeninformationssystem
LaFIS-LFK	Landwirtschaftliches Feldblock Kataster
LAN	Local Area Network
LE	Landschaftselement
LPIS	Land Parcel Identification System
M	
MLUV	Ministerium für Ländliche Entwicklung, Umwelt und Verbraucherschutz (Brandenburg)
MVC	Model View Controller
N	
NEA	Non Eligible Area (= Nicht förderfähige Fläche)
O	
OGC	Open Geospatial Consortium
OSI	Open Systems Interconnection
P	
PB	Physical Block (= Feldblock)
POM	Project Object Model
R	
RDBMS	Relationales Datenbankmanagementsystem
S	
SOAP	Simple Object Access Protocol
SOC	Server Object Container
SOM	Server Object Manager
SVG	Scalable Vector Graphics

T	
t.B	Topologische Beziehung
TCP/IP	Transmission Control Protocol/Internet Protocol
TK25	Topographische Karte im Maßstab 1:25.000
U	
UML	Unified Modeling Language
UMN	University of Minnesota
URL	Uniform Resource Locator
W	
W3C	World Wide Web Consortium
WAN	Wide Area Network
WFS	Web Feature Service
WFS-T	Web Feature Service transactional
WMS	Web Map Service
WPS	Web Processing Service
WS	Web Service
WSDL	Web Service Description Language
WWW	World Wide Web
X	
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language

1 EINFÜHRUNG

1.1 PROBLEMSTELLUNG UND MOTIVATION

Im Zuge der *Gemeinsamen Agrarpolitik (GAP)* richtete die EU ein Förder- und Kontrollsystem für landwirtschaftliche Flächen ein, das *integrierte Verwaltungs- und Kontrollsystem (InVeKoS)*. Hierbei kommen unter anderem GIS- und Fernerkundungsmethoden zur Flächenkontrolle und Identifizierung von landwirtschaftlich genutzten Parzellen zum Einsatz. (Krause, 2006)

Die Firma *GAF AG (Gesellschaft für angewandte Fernerkundung)* entwickelte hierzu für die Verwaltung von landwirtschaftlichen Flächen in Deutschland das *Landwirtschaftliche Flächeninformationssystem (LaFIS)*. Auf Grundlage der bestehenden Anwendung *LaFIS-LFK (Landwirtschaftliches Feldblockkataster)* soll nun ein Prototyp für ein *Land Parcel Identification System (LPIS)* mit neuer Technologie mit einem Web-Client basierend auf *ArcGIS Server (AGS)* konzipiert und entwickelt werden. Dabei dienen Daten vom brandenburgischen *Ministerium für Ländliche Entwicklung, Umwelt und Verbraucherschutz (MLUV)* als Testdaten.

Der zu entwickelnde Prototyp LaFIS-Web soll aufzeigen, wie die Entwicklung einer browserbasierten Web-Editing Anwendung mit AGS Technologie umsetzbar ist und wie sich Prüfungen von topologischen Beziehungen darin realisieren lassen. Die topologischen Beziehungen aus LaFIS LFK flossen in die Spezifikationen für die LaFIS-Web Anwendung in überarbeiteter Form ein.

Das Editieren in Web-GIS Anwendungen findet in den letzten Jahren immer häufiger Anwendung, dank frei verfügbarer Bibliotheken, die u.a. auf den *Open Geospatial Consortium (OGC)* Standards *Web Feature Service transactional (WFS-T)* und *Geographic Markup Language (GML)* beruhen, sowie proprietärer Lösungen. Der Aspekt des topologisch richtigen Editierens ist jedoch in diesen Lösungen meist nicht zu finden und nur mit großem Aufwand oder mit weiteren OGC Services (z.B. Web Processing Service, WPS) umsetzbar. Aber wie hoch ist der Aufwand mit der proprietären Software ArcGIS Server von ESRI, derartige Funktionen in eine Web-GIS Anwendung zu implementieren? Darüber soll die Arbeit einen Überblick geben und einen Einstieg in die Thematik aufzeigen.

1.2 ZIELSETZUNG

Ziel der Arbeit ist es einen Prototyp eines browserbasierten Web-GIS Clients zu erstellen, der Editierfunktionen zur Erfassung und Verwaltung von landwirtschaftlichen Flächen enthält. Da für Verwaltungsbehörden ein konsistenter Datenbestand von grundlegender Bedeutung ist, soll die Erfassung der Daten unter Berücksichtigung von topologischen und räumlichen Beziehungen erfolgen. Dafür gilt es Methoden zu finden, mit denen die Erstellung eines topologisch konsistenten Datensatzes möglich ist.

Die Umsetzung der Web-GIS Applikation erfolgt auf Basis der ArcGIS Server Technologie. Diese Arbeit untersucht die zur Verfügung stehenden Frameworks und Bibliotheken dahingehend, wie sich mit diesen Werkzeugen Web-GIS Editing Anwendungen erstellen und topologische Prüfungen durchführen lassen können. Die Benutzer dieser Anwendung verfügen normalerweise über keine tiefer gehenden GIS Kenntnisse. Deswegen ist es unerlässlich, die Applikation einfach und benutzerfreundlich zu gestalten.

1.3 LÖSUNGSANSATZ UND STRUKTUR

Die Arbeit beschäftigt sich zunächst mit den Grundlagen von Web-GIS Anwendungen, die sich mit den Web Technologien des W3 Konsortiums und aktuellen Web-GIS Systemen beschäftigen. Weiter werden die Begriffe Topologie und topologische Beziehungen näher betrachtet und hinterfragt in wie weit dies schon in Web-GIS Anwendungen zu finden ist.

Das darauf folgende Kapitel erläutert die ArcGIS Server Technologie mit ihren Tools, Funktionen und Frameworks. Hier werden zudem die Entwicklungswerkzeuge und Tools zur Weiterentwicklung der Web-GIS Anwendung vorgestellt. Darauf bauen die konzeptionellen Ansätze zur Implementierung der topologischen Fragestellungen auf, welche einerseits auf der Erweiterung des AGS Edit Tasks und anderseits auf die Entwicklung eines Topologie Dialoges wie in ArcMap basieren. Der gewählte Lösungsansatz ist zusätzlich mit beschreibenden Anwendungsfällen dargestellt.

Das Kapitel Implementierung beschreibt die Erstellung einer Web-GIS Editing Anwendung mit der ArcGIS Servers Manager Applikation. Darauf folgt die Integration des daraus resultierenden Web Projektes in eine Entwicklungsumgebung zur Weiterentwicklung der Anwendung und Integration der topologischen Fragestellungen

durch Erweiterung des Editor Tasks. Die Implementierung wird mit verschiedenen *Unified Modelling Language (UML)* Diagrammen anschaulich beschrieben.

Abschließend werden die Ergebnisse des browserbasierten Web-GIS Clients und der Erweiterung des Edit Tasks dargestellt und ihre Limitationen erläutert. Den Abschluss bildet ein Ausblick auf die noch zu leistenden Arbeiten, die zur Realisierung für ein Produktivsystem notwendig sind.

1.4 ABGRENZUNG UND PUBLIKUM

Diese Master Thesis zeigt konzeptionelle Ansätze zur Integration von Prüfungen von topologischen und räumlichen Beziehungen in einer Web-GIS Anwendung auf. Bei dieser Arbeit handelt es sich um eine Prototyp-technische Umsetzung bei der beispielhaft ein Implementierungsansatz dieser Prüfungen aufgezeigt wird. Es handelt sich hier jedoch nicht um die Umsetzung einer Endanwendung zur Verwaltung von landwirtschaftlichen Flächen.

Die Arbeit richtet sich an GIS Fachleute und Geo-Informatiker mit Kenntnissen in Geoinformationssystemen und Web-GIS Anwendungen, sowie mit Programmiererfahrungen von Java Web-Anwendungen.

2 GRUNDLAGEN

2.1 VERWALTUNG LANDWIRTSCHAFTLICHER FLÄCHEN

2.1.1 INVEKOS

Im Rahmen der *GAP* der Europäischen Union (EU) wurde ab 1992 aufgrund der Verordnungen (EWG) Nr.3508/92 und (EWG) Nr.3887/92 in den Mitgliedsstaaten das *Integrierte Verwaltungs- und Kontrollsystem (*engl.: *IACS* für *Integrated Administration and Control System)* zur Verwaltung und Kontrolle von flächenbezogenen Agrarsubventionen gefordert und eingeführt. (Krause, 2006)

Dieses Kontroll- und Verwaltungssystem bezog sich auf alphanumerische Daten in einer informatisierten Datenbank und auf analoge Katasterpläne und -unterlagen. Zum Teil wurden bereits Luft- oder Satellitendaten als Basis für den räumlichen Bezug und zur Kontrolle eingesetzt. Für die Antragsstellung versendeten die Landwirtschaftsbehörden erstellte Formblätter an die Betriebsinhaber (Landwirte), welche Angaben zu beantragten Flächen und Katasterinformationen enthielten. (Krause, 2006)

Bei diesem Verfahren stellte man jedoch im Laufe der Jahre einige Schwachstellen fest, z.B. war die eindeutige Identifikation von landwirtschaftlichen Flächen nicht immer gegeben. So legte die GAP mit der Verordnung (EG) Nr. 1593/2000 des Rates fest, dass in den Mitgliedstaaten bis zum Jahre 2005 ein geographisches Flächeninformationssystem aufzubauen und einzuführen ist. Diese Verordnung enthielt im Wesentlichen die Anordnung zum Einsatz von *geographischen Informationssystemen (GIS)* und eine dringliche Empfehlung zum Einsatz von Fernerkundungsdaten. Diese Reform förderte die Entwicklung von *Land Parcel Identification Systems (LPIS)*, die heute ein zentraler Bestandteil von InVeKoS sind und im deutschsprachigen Raum als *InVeKoS-GIS* (engl.: *IACS-GIS*) bezeichnet werden. (Krause, 2006)

InVeKoS-GIS soll eine zweifelsfreie und eindeutige Identifikation von landwirtschaftlichen Parzellen vereinfachen bzw. ermöglichen. Dazu ist die Erstellung eines Referenzsystems erforderlich, dessen Aufbau sich nach (Krause, 2006) abhängig von den Referenzflächen folgendermaßen unterscheiden kann:

- **Flurstück-System** (Cadastral Parcels)
  Hierbei dienen amtliche Flurstücksgrenzen als Grenzen für landwirtschaftliche Flächen.
- **Schlag-System** (Agricultural Parcels)
  Ein Schlag ist eine zusammenhängende landwirtschaftliche Fläche, die von einem Betriebsinhaber genutzt wird und mit einer Kulturart bestellt, stillgelegt oder aus der Produktion genommen ist.
- **Feldblock-System** (Physical Blocks, PB)
  Ein Feldblock ist eine zusammenhängende landwirtschaftliche Fläche, die von dauerhaften Grenzen umgeben ist. Die Fläche kann einen oder mehreren Betriebsinhabern gehören und mit einer oder mehreren Kulturarten bestellt sein.
- **Feldstück-System** (Farmers Blocks)
  Ein Feldstück ist eine zusammenhängende landwirtschaftlich genutzte Fläche eines Betriebsinhabers, die mit einer oder mehreren Kulturarten bestellt sein kann.
- Eine **Kombination** der 4 oben genannten Systeme ist ebenfalls möglich

In Brandenburg handelt es sich um das Feldblock-System.

Die Verordnung (EG) Nr. 1782/2003 legte die Entkopplung der Direktzahlungen von der Produktion und die Bindung an Umweltauflagen fest. Die auch unter den Namen *Cross Compliance* bekannte Reform fordert u.a. die Aufnahme von *Landschaftselementen (LE)* in das InVeKoS-GIS Referenzsystem. Landschaftselemente sind traditioneller Bestandteil guter landwirtschaftlicher Anbau- und Nutzerpraktiken und haben aus ökologischer Sicht eine herausragende Bedeutung für Artenvielfalt in der Agrarlandschaft. Sie bieten Lebensräume für wildlebende Tier- und Pflanzenarten und stellen eine Bereicherung für das Landschaftsbild dar. Landschaftselemente unterscheiden sich in Cross Compliance relevante LEs, die erfasst und bei der Antragsstellung verpflichtend anzugeben sind, und sonstige LEs, die beantragt werden können. (Krause, 2006)

2.1.2 LaFIS

Für die Verwaltung von landwirtschaftlichen Flächen entwickelte die GAF AG die *LaFIS* (Abkürzung für *Landwirtschaftliches Flächeninformationssystem*) Softwareprodukte. Das *LaFIS-LFK (Landwirtschaftliches Feldblockkataster)* ist Teil dieser Produktfamilie, bei

der es sich um eine Client-Server Anwendung mit jeweils einer zentralen Geo- und Sachdatenbank handelt, die über eine Internet- oder Netzwerkverbindung angebunden ist. Es ist sowohl die Verarbeitung von Vektor- als auch von Rasterdaten möglich. Beim Client handelt es sich um einen so genannten *Fat Client*, der auf dem Anwender Rechner installiert werden muss.

Die LaFIS-LFK Anwendung dient zur Erstellung und Pflege eines landwirtschaftlichen Referenzsystems (Flächenkataster), was in verschiedenen Datensätzen erfolgt. Die Darstellung dieser geographischen Daten im LaFIS-LFK ist nach dem Ebenenprinzip aufgebaut und in Ebenengruppen (Group Layers) und Ebenen (Layer) gegliedert. Den Ebenen-Aufbau und die einzelnen Layer zeigt die Abbildung 2-01.

Abb. 1: Ebenenstruktur in LaFIS-LFK

Die Bearbeitung der geographischen Daten läuft dabei folgendermaßen ab: Ist ein Feldblock oder Landschaftselement aus der Referenzebene zu ändern, so wird das Element in die entsprechende Bearbeitungsebene kopiert. In den Bearbeitungsebenen erfolgt dann die Aktualisierung der Geometrie- oder Sachdaten oder die Neudigitalisierung eines Elements. Die geänderten oder neu erstellten Elemente bleiben bis zum Ende des Jahres in den Bearbeitungsebenen, woraus dann nach einem Antragsjahr die beiden Referenzebenen neu generiert werden. Diese sind dann Grundlage für die Subventionsanträge der Landwirte.

Die Entwicklung des Prototyps konzentriert sich auf das Arbeiten in der Bearbeitungsebene, also auf die Digitalisierung neuer bzw. Änderung vorhandener Elemente.

2.2 WEB-GIS UND WEB-MAPPING

Die Darstellung von Geoinformationen im Internet gewinnt immer mehr an Bedeutung, vor allem seit Anwendungen wie Google Maps auf dem Markt erschienen sind. Im Jahre 1994 erschien in den USA die erste experimentelle Webseite zur Darstellung einer Weltkarte zur interaktiven Informationsabfrage. Diese Applikation gilt als Urform aller netzbasierten Kartenserver und stellte den Beginn der jüngsten technologischen Wandlung der Kartographie dar (Asche, 2001). Diese Art von Anwendungen beschränkten sich auf das Bereitstellen von dynamisch erzeugten georeferenzierten Dokumenten, z.B. mittels Common Gateway Interface (CGI) Schnittstellen, die meist auf einen bestimmten Datensatz abgestimmt waren. Später wurden, basierend auf Map Servern geschlossener Systeme, funktionsreichere Webanwendungen entwickelt, die jedoch eine Erweiterung des Browsers durch Plug-ins erforderten. Eine Herausforderung und vor allem das Ziel des Web-GIS von heute ist es, die geographische Informationsverarbeitung in das WWW zu transferieren und zu integrieren. (Strobl, 2001) Heute bieten die Informations- und Telekommunikationstechnologien (ITK) des Web 2.0 immer mehr Möglichkeiten leistungsfähige Web-Mapping Anwendungen zu erstellen.

2.2.1 GRUNDLAGEN UND AUFBAU DES WEB-GIS

Das Web-GIS ist grundsätzlich nach dem *Client-Server Prinzip* aufgebaut. Der Server stellt dabei Services oder Daten zur Verfügung, welche Clients nutzen, indem sie *Anfragen (request)* an den Server stellen. Dieser beantwortet die Anfragen des Clients und sendet die *Rückantworten (response)* zurück an den Client. Beide kommunizieren über ein festgelegtes Protokoll (z.B. TCP/IP). Die Interaktion zwischen den beiden Einheiten ist *synchron*, wenn der Client nach dem Senden der Anfrage auf die Antwort wartet. Wenn der Client nach dem Absenden jedoch weiter arbeitet, spricht man von einer *asynchronen* Interaktion. Dies hat den Vorteil, dass die Clientanwendungen effizienter arbeiten können, jedoch ist die Implementierung durch eine erhöhte Kontrollkomplexität der Rückantworten schwieriger. (Bengel, 2004)

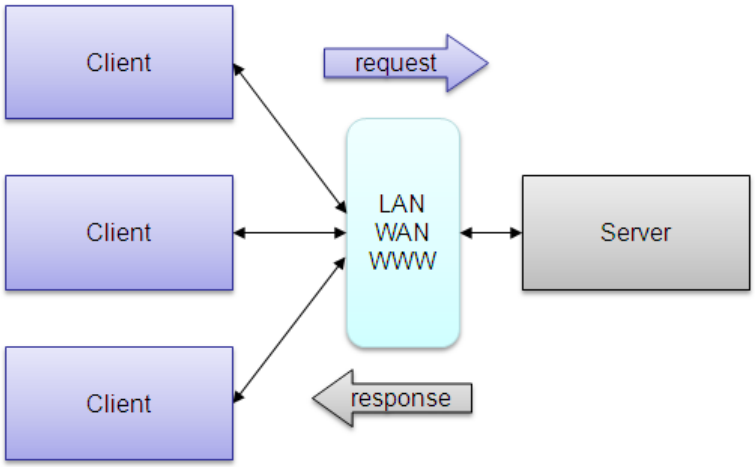


Abb. 2: Client-Server-Achitektur

Eine Web-GIS Anwendung ist auf verschiedene Art und Weise umsetzbar. Die eingesetzten Technologien sind dabei entscheidend, da von ihnen wesentliche Funktionalitäten abhängig sind (Korduan/Zehner, 2008).

Im Internet finden sich heutzutage bereits eine ganze Reihe von Web-GIS Anwendungen, die sich in der Art der Anwendung, Funktionalität und Technologie unterscheiden. Nach (Korduan/Zehner, 2008) lassen sich die *Anwendungsbereiche* des Internet-GIS grob in vier Kategorien einteilen:

- **Einfache Auskunftssysteme**
  Einfache statische und interaktive Anwendungen, in denen Karten mit Sachdaten über Browser oder Plug-in darzustellen sind.
- **Spezialisierte georeferenzierte Auskunftssysteme**
  Hier sind dynamische Systeme notwendig die zusätzliche Dienste bereitstellen, welche der Server während der Anforderung bearbeitet, z.B. Routensysteme.
- **Internetbasierte Geoinformatik-Clients**
  Hierbei kann auf eine zentrale Ressource zugegriffen werden, die erweiterte Funktionalitäten besitzt. Somit sind Analysen, Verschneidungen, Exporte und Änderungen im Sach- und Geometriedatenbestand möglich. Diese Form könnte z.B. eine Alternative zu einer Desktop-GIS Anwendung darstellen.
- **Geodatenportale**
  Sie stellen Geodaten meist von unterschiedlichen Geodatenservern über eine Internetanwendung zur Verfügung oder verkaufen diese.

Bei Internet-GIS Anwendungen sind *Funktionalitäten* wie Präsentieren, Analysieren, Gestalten, Vergleichen, Berechnung oder Abfragen relativ geläufig. Funktionalitäten wie Erfassen, Verändern, Transformieren, Archivieren, Prüfen und Sichern sind jedoch relativ selten und lassen sich nur mit einen erhöhten technischen Aufwand realisieren. Für Internet-GIS Anwendungen sind aber noch weitere, teils nicht GIS-Funktionalitäten von großer Bedeutung, wie z.B. Autorisierung, Authentifizierung, Verwaltung, Abrechnungssystem. (Korduan/Zehner, 2008)

Wie sich Internet-GIS Anwendungen implementieren lassen und welche Komponenten dabei zum Einsatz kommen, zeigt folgende Tabelle als Übersicht:

	Server	Client	Bemerkungen
Statische Webseiten	HTML-Dokumente, Grafikobjekte	Webbrowser	Bewährte und leicht realisierbare Lösung
Interaktive Webseiten	HTML-Dokumente, Grafikobjekte, umfangreiche eingebettete Funktionalitäten	Webbrowser mit Skriptunterstützung	Notwendig bei größeren Datenbeständen und wechselnden Anzeigen
Client-Anwendungen	HTML-Dokumente mit eingebetteter Softwarekomponente, Datendateien	Webbrowser mit Plug-in wie SVG, Java, spezielle Viewer	Bei hohen Ansprüchen an die Darstellung, Skalierbarkeit und Interaktion
Dynamische Webseiten	Dynamische Serveranwendung, Direktzugriff auf Geodaten	Webbrowser	Unabhängigkeit von Client, große Datenbestände und vielfältige Funktionen
Geodateninfrastruktur	Datendienste, die auf vordefinierten Schnittstellen Geodaten bereitstellen	Geodatenclient, z.B. Viewer, Desktop-GIS, Webportal	Zugriff auf verteilte Geodatenbestände
Terminalserver	Hochleistungsserver mit Terminalsuite, GIS-Software und -Daten	Spezieller Client	Voller Zugriff auf ein Desktop-GIS über das Internet

Tab. 1: Technische des Varianten Internet-GIS [1]

2.2.2 INTERNETTECHNOLOGIEN

2.2.2.1 IT-Standards und Web Technologien

Das Internet und *World Wide Web (WWW)*, sowie deren Technologie, bilden die Grundlage für jede Art von Web-GIS Anwendungen. Diese basieren, wie auch das Internet, auf dem Client-Server Prinzip, das auf dem *Transmission Control*

[1] Quelle: (Korduan/Zehner, 2008) Seite 12

Protocol/Internet Protocol (TCP/IP) aus dem *OSI (Open Systems Interconnection)-Referenzmodell* aufbaut. Dieses Referenzmodell beschreibt den physischen und technischen Zugang zum Netzwerk, die Adressierung der Netzwerkteilnehmer und Austausch der Datenpakete. In ihm werden auch die Überwachung des Datenstroms und die Verwaltung des Datenverkehrs beschrieben, sowie die Informationsbereitstellung und Nutzung für die Darstellung. Der folgende Abschnitt bietet einen Überblick über die WWW Technologien, die im Rahmen dieser Arbeit Verwendung finden.

URL steht für *Uniform Ressource Locator* und wird umgangssprachlich als Internetadresse bezeichnet. Über Netzwerkprotokolle (z.B. HTTP, HTTPS oder FTP) erfolgt die Lokalisierung und Identifizierung von Ressourcen im Internet.

HTTP *(Hypertext Transfer Protocol)* ist ein Protokoll zur Übertragung von Daten über ein Netzwerk. Das Transportprotokoll ist hauptsächlich TCP. Mit den HTTP-Request Methoden Get und Post besteht die Möglichkeit Argumente zu übertragen. Bei der Get Methode werden die Argumente als Parameter-Werte-Paare nach dem Zeichen „?“ an die URL angehängt und auch so übergeben, sie sind somit sichtbar. In der Post-Methode sind die Argumente nicht sichtbar, die Übertragung erfolgt in den HTTP-Kopfdaten, so dass hier größere Datenmengen gegenüber der Get-Methode möglich sind. HTTP ist ein zustandsloses Protokoll, jedoch können über so genannte Cookies in den Header-Informationen Anwendungen mit Statusinformationen realisiert werden. (Wikipedia 1, 2009)

HTTPS steht für *Hypertext Transfer Protocol Secure* und ermöglicht im Gegensatz zu HTTP eine sichere Übertragung von Daten mit Hilfe von SSL/TLS (Secure Socket Layers/Transport Layer Securtiy).

HTML steht für *Hypertext Markup Language* und ist eine sogenannte Auszeichnungssprache. Deren Aufgabe ist es, logische Bestandteile (wie z.B. Überschriften, Textabsätze, Listen, Tabellen oder Grafikreferenzen) eines textorientierten Dokumentes zu beschreiben und Verweise (Hyperlinks) einzubinden. Durch die stetig wachsende Bedeutung von XML wurde HTML basierend auf XML neu definiert und ist mittlerweile als Standard in den zwei Versionen HTML und XHTML verfügbar (SELFHTML 1, 2009). Auch Web-GIS Komponenten lassen sich in ein HTML Dokument einbinden und zusammen mit anderen Inhalten im Web darstellen (Korduan/Zehner, 2008).

CSS ist die Abkürzung für *Cascading Stylesheets*. Sie ist vom W3C normiert und eine Ergänzungssprache zu HTML und XHTML, mit der sich Formateigenschaften, wie z.B. Schriftart und Schriftgröße, und Positionsangaben einzelner HTML Elemente genau definieren lassen. Ein wesentlicher Vorteil von CSS ist die Möglichkeit, zentrale Formate in einer separaten Datei zu definieren, die von jedem HTML Dokument im Projekt referenziert und verwendet werden können. (SELFHTML 2, 2009)

XML *(Extensible Markup Language)* ist ein Standard des W3C und ist eine Metasprache zur Definition beliebiger Auszeichnungssprachen (z.B. Scalable Vector Graphics oder XHTML), die dann wiederum strukturierte Daten in XML-Dokumenten beschreiben. Sogenannte *Tags* definieren dabei Elemente, denen Attribute und Werte zugewiesen werden können. Welche Elemente in welcher Reihenfolge definierbar sind, wird in so genannten XML-Dokumentenformaten festgelegt. Dies sind entweder *Document Type Definition (DTD)* oder XML-Schema Dateien. (SELFHTML 3, 2009)

JavaScript *(JS)* ist eine eigene Programmiersprache mit dessen Hilfe sich Web-Seiten optimieren lassen. Sie ist sozusagen eine Ergänzung zu HTML. JavaScript wird vom Browser interpretiert und somit clientseitig eingesetzt. Mit JavaScript ist die Erstellung von interaktiven und dynamischen Webseiten möglich. JS ist nicht vom W3C standardisiert, setzt aber das DOM des W3C um. (SELFHTML 4, 2009)

Das ***DOM*** *(Document Object Model)* wurde auch vom W3C-Konsortium standardisiert. Das DOM repräsentiert die Daten und Inhalte eines HTML- oder XML-Dokumentes und ist eine Schnittstelle für den Zugriff auf Objekte im Dokument. Dies ermöglicht eine dynamische Manipulation von HTML- und XML-Dokumenten. Die Funktionen zur Manipulation des DOM sind in vielen Programmiersprachen implementiert, so auch in JavaScript. (SELFHTML 4, 2009)

DHTML steht für *dynamic HTML* und ist keine neue Sprache oder eine klassische HTML Erweiterung. Vielmehr ist es ein Sammelbegriff für verschiedene Lösungen um Elemente einer Web-Site dynamisch während der Anzeige zu ändern. Ein Lösungsansatz wäre z.B. das Ändern oder Hinzufügen von HTML Inhalten mit JavaScript über das DOM, ohne dass dabei die Seite vollständig neu geladen werden muss. (SELFHTML 5, 2009)

AJAX steht für *Asynchronous JavaScript and XML* und beschreibt ein Programmierungs-modell für Web-Anwendungen mit asynchroner Datenübertragung zwischen Server und

Browser. Dadurch ist es möglich, nur Teile der Inhalte einer Webseite zu aktualisieren. Es ist nicht nötig die komplette Web-Seite vom Webserver neu zu laden. Mit Hilfe der AJAX Technologie ist es möglich Desktop ähnliche Web-Anwendungen zu erstellen. Durch die JavaScript-Funktion XMLHttpRequest werden im Hintergrund einer Web-Applikation Anfragen an den Server gesendet. Die Antwort erhält der Client im XML Format und das DOM bindet diese in die Anwendung ein ohne die Seite neu zu laden. (Korduan/Zehner, 2008)

Ein ***Web Service*** *(WS)* ermöglicht den Zugriff auf Anwendungsfunktionen über Schnittstellen, die mittels TCP/IP zugänglich sind. Das *Simple Object Access Protocol (SOAP)* überträgt die Daten über HTTP-POST zwischen Client und Server. SOAP basiert auf XML und arbeitet wie auch HTTP nach dem Request/Response Prinzip. SOAP befindet sich derzeit in der Version 1.2 und ist ein W3C Standard. Die *Web Service Description Language (WSDL)* basiert auf XML und beschreibt die Schnittstelle für den Web Service. Das WSDL Dokument spezifiziert die Funktionen, Protokolle und Adresse des Web Services. (Korduan/Zehner, 2008)

2.2.2.2 Web Technologien mit Java

Java ist eine objektorientierte Programmiersprache auf Basis von OpenSource, die auf der Java-Technologie aufbaut und eine Vielzahl von Bibliotheken und Frameworks zur Softwareentwicklung bereitstellt. Zur Laufzeit wird der kompilierte Java Bytecode von einer virtuellen Maschine (JVM) interpretiert. Die JVM ist in der Java Runtime Environment (JRE) enthalten und steht für verschiedene Plattformen zur Verfügung. Dies erlaubt es Java Programme auf unterschiedlichen Plattformen auszuführen, sofern die JVM/JRE installiert ist. Zu den Standard-Bibliotheken gibt es für nahezu alle Aufgaben kommerzielle oder quelloffene Bibliotheken. (Krüger, 2007)

Java ist besonders für die Entwicklung von Enterprise- und Web-Anwendungen geeignet, da es plattformunabhängig ist, zudem viele Technologien und eine große Menge an Frameworks und Bibliotheken bietet. Die Basis für Java Webapplikationen sind die Java Technologien *Java Servlets* und *JavaServer Pages (JSP)*.

Servlets sind Java-Objekte in einem Server, die Anfragen von Clients aus dem Internet bearbeiten und beantworten. Die Hauptfunktionalität der Servlets steckt in der `doGet()`

Methode, in der die Antwort mit der Methode `println()` generiert wird, also Webseiten im HTML Format für den Client erstellt und an diesen gesendet werden. Servlets laufen in so genannten Web Containern und sind meist im Kontext des Webservers zu finden. (Farley/Crawford, 2005)

***JavaServer Pages** (JSP)* ist Bestandteil der Java Enterprise Spezifikation Java EE 5. JSP Seiten sind im Grunde HTML-Seiten mit eingebettetem Java-Code. Es können aber auch einfache HTML-Seiten ohne Java-Code bereits JSP Seiten sein. JSP bedient sich der Servlet Technologie, indem es bei Anfragen die entsprechenden JSP Seiten in Servlets umwandelt, diese vom Webserver als solche verarbeitet werden und das Ergebnis der Anfrage so in der JSP Seite dargestellt wird. JSP-Seiten befinden sich, wie auch die Servlets und HMTL Seiten, in Web Containern des Applikationsservers. (Farley/Crawford, 2005)

JavaServer Faces *(JSF)* basieren auf der Servlets- und JSP-Technologie und sind Bestandteil der Java EE 5 Web Technologie. JSF ist ein serverseitiges standardisiertes Framework zur Entwicklung von Web basierten User-Interfaces, der Präsentationsschicht von Webanwendungen. Die wieder verwendbaren UI-Komponenten lassen sich einfach in Web-Seiten einbinden, an Datenquellen anbinden und Interaktionen mit clientseitigen Events und serverseitigen Event-Handlern prozessieren. Das JSF-Framework verfolgt eine strikte Trennung zwischen Anwendungslogik und GUI-Darstellung (Graphic User Interface) und unterstützt somit die Applikationsentwicklung nach dem *Model-View-Controller (MVC)* – Prinzip. (Farley/Crawford, 2005)

2.2.2.3 OGC Standards

WMS – Web Map Service

Der OGC Web Map Service (WMS) erlaubt einem Nutzer, mittels parametrisierter Anfragen geographische Informationen im Raster-Grafikformat von einem WMS Server abzufragen. Der WMS Server ist für die graphische Aufbereitung der Geodaten verantwortlich, die sowohl Raster- als auch Vektordaten sein können. Die Anfragen des Clients erfolgen über HTTP Anfragen, hierbei werden folgende Operationen von einem OGC konformen WMS unterstützt:

- *GetCapabilities:* Frägt die Fähigkeiten des Services ab
- *GetMap:* Gibt ein georeferenziertes Rasterbild (im .jpg, .png, … Format) zurück

- *GetFeatureInfo (*optional): Liefert thematische Informationen zu geographischen Daten an einer bestimmten Koordinate

(Wikipedia 2, 2009)

WFS – Web Feature Service

Der *Web Feature Service (WFS)* ist ein OGC Standard, der einem Nutzer den Zugriff auf geographische Daten (Features) über das Internet erlaubt. Er ermöglicht geographische Features im Vektorformat mit Hilfe der Geography Makup Language (GML) abzufragen. Die Anfragen des Client werden über das Protokoll HTTP zum WFS Server gesendet, dieser sendet die Antwort in Form von XML Dokumenten zurück. Der WFS Standard unterstützt folgende Operationen:

- *GetCapabilities:* Abfragen der Fähigkeiten des Services
- *DescribeFeatureType:* Gibt Informationen über die Struktur eines einzelnen Features
- *GetFeature:* Gibt eine spezifiziertes Feature zurück
- *GetGMLObject:* Gibt als Ergebnis eine GML Datei zurück

(Wikipedia 3, 2009)

WFS-T – Web Feature Service Transactional

Dieser Standard unterstützt alle Operationen des WFS Standards plus die beiden weiteren Operationen

- *Transaction:* Ermöglicht Features zu ändern oder neu zu erstellen
- *LockFeature:* Sperrt das Feature während der Transaktion für andere Benutzer

Durch die Transaction Operation ist das Aktualisieren, Löschen und neu Anlegen von geographischen Features möglich. (Wikipedia 3, 2009)

2.2.3 Vor- und Nachteile von Web-GIS

In 2.2.1. wurden bereits verschieden Arten von Web-GIS Anwendungen vorgestellt. Je nach Kategorie, Variante und eingesetzten Technologien haben diese verschiedene Vor- und Nachteile:

Eine Web-GIS Applikation bietet folgende *Vorteile*:

- Eine Web-GIS Applikation unterstützt eine große Anzahl von Nutzern.
- Die Anwendung kann von jedem Rechner mit Internetverbindung verwendet werden, der einen Browser installiert hat.
- Der Client Rechner benötigt nur geringe technische Anforderungen.
- Web-GIS Anwendungen stellen einen vereinfachten Zugang zu raumbezogenen Informationen und Geodaten dar.
- Auf der Nutzerseite entfallen meist die Software- und Lizenzkosten oder sind sehr gering.
- Standardisierungen in der Informations und Kommunikations (IuK) Technologie vereinfachen den Austausch der Geodaten übers Internet.

(Korduan/Zehner, 2008)

Neben den vielen Vorteilen haben Web-GIS Anwendungen aber auch gewisse *Nachteile*:

- Die Web-GIS Anwendung kann durch den Nutzer weniger angepasst werden; es stehen nur die vorgegebenen Daten und Funktionalitäten zur Verfügung.
- Die Umsetzung von komplexeren Funktionalitäten auf der Nutzerseite ist nur mit besonderen Lösungen umsetzbar, da hierfür die nötigen Standards noch fehlen.
- Die verschiedenen technischen Lösungen verkomplizieren derzeit den Sachverhalt
- Es sind besondere Sicherheitsüberprüfungen für Web-GIS Anwendungen notwendig.

(Korduan/Zehner, 2008)

2.2.4 Aktuelle Technologien für Web-GIS

Zu den bekanntesten Vertretern von Web-GIS Anwendungen gehören die kostenlosen Web-Mapping Anwendungen wie Google Maps oder Microsoft Bing (früher Virtual Earth). Sie bieten jeweils zusätzlich *Application Developer Frameworks (ADFs)* für die Integration von Karten in Webseiten und für andere individuelle Anpassungen an.

Neben diesen populären Anwendungen gibt es für die Erstellung und Entwicklung von Web-GIS Anwendungen eine Reihe von Produkten aus Open Source Projekten und von kommerziellen Anbietern, von denen im Folgenden eine Auswahl erwähnt wird.

Im *Open Source* Bereich sind für die Erstellung von clientseiter Komponenten vor allem die OSGeo Projekte *OpenLayers*, *Mapbender* und *MapGuide OS* zu nennen. An der Universität Rostock ist *kvwmap* als ein weiteres Web-GIS Framework entstanden. Auf der Server Seite haben sich der *(UMN) MapServer* und *GeoServer* etabliert.

Die großen kommerziellen Anbieter offerieren neben ihren Server Komponenten weitere Tools, Frameworks und APIs zur Erstellung von Web-GIS Applikationen. Intergraph führt in ihrem Portfolio das Produkt GeoMedia WebMap. Die Firma Autodesk präsentiert sich mit Autodesk MapGuide Enterprise als Web-GIS Lösung am Markt, wofür auch eine Open Source Version verfügbar ist. ESRI bietet als Server Komponenten ArcIMS und den ArcGIS Server an, zu dem mehrere APIs für Web-GIS Anwendungsentwicklung zur Verfügung gestellt werden. Das dritte Kapitel stellt den ArcGIS Server und das Web ADF näher vor

2.3 Räumliche Beziehungen

2.3.1 Topologie

Topologie ist ein eigenständiger Bestandteil der Mathematik und beschäftigt sich heutzutage hauptsächlich mit nichtmetrischen räumlichen und strukturellen Beziehungen beliebiger Elemente in abstrakten Räumen (Bill, 1999). Sie spielt in der Datenmodellierung in GI-Systemen eine große Rolle. Man unterscheidet:

Algebraische Topologie: Klärung von Fragen des euklidischen Raumes (z.B. Verschlingungen und Verkettungen von Knoten und Kanten) mit algebraischen Hilfsmitteln.

Mengentheoretische Topologie: Untersuchung von speziellen Abbildungen in allgemeinen Räumen, um Klassifizierungen von Figuren zu ermöglichen.

Die Bestimmung von topologischen Invarianten, wie Geschlossenheit, Schnittpunkttreue, Trennung innen/außen und Randpunkteigenschaften, ist die Aufgabe der Topologie. Mit ihrer Hilfe lassen sich in GI-Systemen verschiedene Problemstellungen, wie Konsistenzprüfungen, Nachbarschaftsbeziehungen und kürzeste Wege, bearbeiten und lösen.

Die geometrischen Grundeinheiten Punkt, Linie und Fläche werden in der Topologie folgendermaßen definiert (nach (Bill, 1999)):

- *Knoten* (engl. Node, Grundeinheit des Vektormodells)
- *Kanten* (engl. Edge, bestehend aus Anfangs- und Endknoten)
- *Maschen* (engl. Face, umgeben von Kanten)

Topologien für geographische Daten können in professionellen Datenbanken gehalten oder von Schnittstellen (z.B. ArcSDE) verwaltet werden, die zwischen der GIS-Anwendung und dem Datenbanksystem stehen. Diese Schnittstellen erweitern meist das relationale Datenbankschema zu einem objektrelationalen Datenbankschema und erleichtern so das Speichern von wenig separablen, bzw. räumlich korrelierenden Geodaten, im Gegensatz zu atomaren (Sach-)Daten. (Bill, 1999)

2.3.2 KONSISTENZBEDINGUNGEN

Es gelten bestimmte Konsistenzbedingungen, um einen Datensatz konsistent zu halten. Eine Definition der Bedingungen ist mit Hilfe der Topologie möglich. Es darf nur ein Knoten an derselben Stelle im Raum vorhanden sein. Knoten werden durch Kanten miteinander verknüpft, wobei die Art der Verbindung (z.B. Gerade oder Kurve) topologisch äquivalent ist. Durch die Definition eines Anfang- und Endknotens besitzen Kanten eine Richtung. Die Eindeutigkeit der Kanten ist genauso wie bei den Knoten gefordert, d.h. sie dürfen nicht übereinander liegen. Kreuzen sich zwei Linien, so entsteht ein Knoten. Die Kanten-Knoten-Struktur entspricht einem Graphen und stammt aus der Graphentheorie. Dabei sind zwei Knoten *adjazent*, wenn diese durch eine Kante verbunden sind. Mehrere Kanten sind *inzident*, wenn sie in einem Knoten beginnen oder enden. (Barthelme, 2005)

Flächen im Vektormodell definieren sich über ihre umgebenden (geordneten und in sich geschlossenen) Kanten (äußerer Rand) und keinen, einen oder mehrere innere Ränder mittels der Kanten-Knoten-Struktur. Die Ränder der Flächen werden oft auch als Ringe bezeichnet. Flächenaussparungen sind als Inseln definiert. Eine Masche kann also durch eine Folge von Ringen repräsentiert werden, die durch geeignete Trennzeichen voneinander abgehoben sind. (Barthelme, 2005)

Sind die topologischen Elemente Knoten, Kanten und Maschen „gleichberechtigt“ auf eine Ebene gestellt und nicht, wie zuvor beschrieben, hierarchisch aufgebaut, gelangt man zu einer Knoten – Kante – Masche Dreierbeziehung, die nach (Barthelme, 2005) folgende Abhängigkeiten bzw. Beziehungen aufweist:

- Jede Kante wird von genau zwei Knoten (Anfang-Ende) begrenzt
- In jedem Knoten können mehrere Kanten beginnen bzw. enden
- Jede Kante hat eine linke und eine rechte Masche, bezogen auf die Fortschreitungsrichtung
- Jede Masche wird von einem äußeren (Kanten-)Ring und fallweise mehreren inneren (Kanten-)Ringen begrenzt

Bezieht das Modell auch isolierte Knoten und ein Außenraum mit ein, so ergeben sich zwei weitere Beziehungen zwischen den topologischen Elementen:

- Jeder Knoten ist Randpunkt mehrerer Maschen – so er nicht isoliert ist
- Jede Masche kann mehrere (innere und äußere) Randknoten haben

Eine Fläche mit zwei Aussparungen kann nach (Barthelme, 2005) demnach wie folgt beschrieben werden:

Fläche = Masche $_1$ – Masche $_2$ – Masche $_3$

Für die Formalisierung von topologischen Eigenschaften eignen sich besonders gut Hilfsmittel aus der Graphentheorie, auf denen die Eigenschaften topologischer Konsistenzbedingungen basieren. Graphen sind entweder *zusammenhängend* oder *nicht zusammenhängend*. Gibt es im Graphen zwischen zwei Knoten mehrere Wege, so enthält der Graph ein oder mehrere *Zyklen*. Enthält er *keine Zyklen*, so spricht man von einem Baum. Man spricht von einem Graphen mit *nur Zyklen*, wenn es zu jedem Knoten mehrere Wege gibt. Außerdem ist ein Graph *planar*, wenn er sich in einer Ebene abbilden lässt, ohne dass sich eine der Kante mit einer anderen Kante schneidet. (Barthelme, N., 2005)

2.3.3 TOPOLOGISCHE BEZIEHUNGEN

Topologische Beziehungen lassen sich gut mit dem 9-Intersection Schema nach Egenhofer klassifizieren. Bei dieser Methode werden von zwei topologischen Elementen (Knoten, Kante oder Masche) das *Innere* A^i, der *Rand* A^r und das *Komplement* A^k (Außenraum) miteinander verglichen und in einer 9-tupel Schnittmenge (siehe unten) dargestellt. Dabei

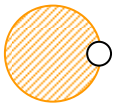
ist zu beachten, dass bei Linien die Anfangs- und Endknoten der Rand und das Innere die Kante ist. Bei Knoten fallen der Rand und das Innere zusammen. Das Ergebnis der Schnittmenge von A und B ist entweder mit leer (0) oder nicht leer (1) angegeben.

9-tupel Schnittmenge:

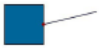
$A^r \cap B^r$ $A^r \cap B^i$ $A^r \cap B^k$
$A^i \cap B^r$ $A^i \cap B^i$ $A^i \cap B^k$
$A^k \cap B^r$ $A^k \cap B^i$ $A^k \cap B^k$

So liefert ein Punkt (A), der eine Fläche (B) berührt, folgendes Ergebnis:

1 0 0
1 0 0
1 1 1



Nach diesem Schema können Knoten, Kanten und Maschen mit- und untereinander verglichen werden. Die häufigsten topologischen Beziehungen sind nach Elementen gegliedert mit graphischen Beispielen in folgender Tabelle dargestellt:

	poly-poly	line-line	point-point	poly-line	poly-point	line-point
Disjoint						
Meet						
Overlap						
Contains						
Inside						
Covers						
Covered by						
Equal						

Tab. 2: Tabelle mit Darstellung der häufigen topologischen Beziehungen [2]

2.3.4 Geometrisch-topologische Methoden (Abfragen)

Aufbauend auf dem Vektormodell und den topologischen Beziehungen sind in GI-Systemen geometrische und topologische Methoden (Abfragen) implementiert. In der Vektorgeometrie lassen sich die Fragestellungen zum Lagevergleich im Prinzip auf ein Grundproblem zurückführen, nämlich auf die Lage eines Punktes bezüglich einer Linie. Darauf aufbauend können Lagevergleiche zwischen Punkten, Linien und Polygonen erstellt und Schnittpunkte gefunden werden. (Barthelme, 2005)

[2] Quelle: www.gitta.info/SpatialQueries/de/text/SpatialQueries.pdf, aufgerufen am 06.05.09

Das menschliche Auge erkennt topologische Unstimmigkeiten sehr gut. Die Umsetzung eines konsistenten Datensatzes in einem GIS ist jedoch alles andere als trivial und fordert eine ganze Menge an Maßnahmen. Um eine topologisch ungeordneten Datensatz (Spaghetti-Code) in einer Geo-Datenbank topologisch richtig abzuspeichern, sind nach (Barthelme, 2005) folgende Maßnahmen notwendig:

- Paarweiser Vergleich aller Spaghetti auf etwaige Schnittpunkte
- Mittelung von Punkten, die innerhalb einer vorgegebenen Toleranz liegen (ESRI)
- Ermittlung von Flächenrändern und Aussparungen, sowie Erkennen von Problemzonen, wo eine Flächenbildung aufgrund von Sackgassen oder zu kurz bzw. zu lang geratenen Linienstücken nicht möglich sind
- Elimination dieser Fehler und erneuter Durchlauf

Für die Erlangung eines topologisch richtigen/konistenten Datensatzes und zur Durchführung oben genannter Maßnahmen, sind im GIS verschiedene Werkzeuge und Funktionen notwendig:

- Verlängern, Kürzen oder Löschen von Kanten
- Aufschneiden von Linien und Setzen von Knoten
- Mitteln von Knotengeometrien

(Barthelme, 2005)

Man spricht von einer *Transaktion*, wenn ein topologisch konsistenter Datenbestand durch eine topologische Operation von einem konsistenten Zustand A in einen ebenfalls konsistenten Zustand B überführt wird. Eine Überwachung derartiger Transaktionen ist mit Hilfe der *Eulerschen Gleichung* möglich, welche die Anzahl der Knoten (Vertices), Kanten (Edges) und der Maschen (Faces) mit der Anzahl der nicht zusammenhängenden Teile (Shapes) in Zusammenhang bringt. Die Gleichung lautet folgendermaßen:

$$\mathbf{V + F = E + S}$$

Diese Art der Überwachung liefert jedoch bestenfalls Verdachtsmomente, die noch manuell zu prüfen sind. (Barthelme, 2005)

Weitere Methoden in der Vektortopologie findet man in topologischen Datenbeständen mit Netzstrukturen, wie z.B. die Findung von optimalen Routen und Touren. Diese sind allerdings nicht Gegenstand dieser Arbeit.

2.3.5 TOPOLOGIE IM WEB-GIS

Web-GIS Anwendungen, die auf einen topologischen Datensatz zugreifen, sind in der Praxis sehr selten. Meist werden Internet-GIS Anwendungen nur zur Visualisierung von Geodaten verwendet. Für derartige Datenbestände ist das Vorhalten einer Topologie nicht zwingend notwendig. Ist die Bearbeitung der Daten erforderlich, ändert sich dies. Um auch nach Editiervorgängen eine konsistenten Datenbestand zu gewährleisten, ist die Definition von bzw. Prüfungen nach topologischen Regeln erforderlich. Bisher gibt es jedoch nur wenige Systeme, die das Editieren mit einem Web-GIS Client unterstützen. Bei diesen ist meist nur das Editieren von Simple Features möglich, welche die Einschränkung haben, dass nur zwei Linien von einem Punkt in einem Linienzug ausgehen dürfen. Durch diesen Umstand ist die Speicherung von Topologien eingeschränkt. Es wird im Simple-Feature Modell jede Masche separat als Polygon abgespeichert. So ist dem Datenmodell nicht bekannt, welche Polygone einen gemeinsamen Knoten teilen. Eine redundante Datenhaltung für Punkte an gleicher Stelle ist die Folge. Eine Änderung des Polygons erfordert aus diesem Grund eine Prüfung, ob auch andere Polygone einen Knoten an selbiger Koordinate haben, der auch mit geändert werden muss. (Korduan/Zehner, 2008)

Für die Umsetzung des topologisch korrekten Editierens in einer Web-GIS Anwendung gibt es einen client- und serverseitigen Ansatz. Beim ersteren wäre es nötig, dass alle Punkte, welche die gleiche Lage des zu verändernden Punktes haben, ebenfalls gleichermaßen verändert werden. Inkonsistenzen, wie gleiche Lage mit bestehenden Punkten oder Überschneidungen von Linien, sind dabei auch clientseitig zu prüfen und zu verhindern. Nach einer Prüfung und Bereinigung werden alle geänderten Features schließlich an den Server geschickt. Bei der serverseitigen Umsetzung wird das geänderte Feature zum Server geschickt und der Server kümmert sich um die Aufrechterhaltung des konsistenten Datensatzes. Schwierigkeiten, die beim Löschen, Teilen oder Vereinen auftreten, sind dabei zu beachten und Regeln für die Verarbeitung der anhängenden Sachdaten zu definieren. (Korduan/Zehner, 2008)

Für die serverseitige Prüfung der Topologie bietet z.B. PostGIS das Topology Modul an. Das Topologieschema ist dabei in mehreren Tabellen gespeichert. In die Datenbank (DB) eingelesene Topologieelemente können validiert, sowie neue Geometrien erzeugt werden. (Korduan/Zehner, 2008)

3 LÖSUNGSANSATZ

Bei der Entwicklung des Prototyps LaFIS-Web ist beim Lösungsansatz zu berücksichtigen, dass die Endanwender keine GIS Fachkräfte mit tiefer gehenden GIS Kenntnissen sind. Daher sollte die Integration der topologischen Aspekte in die Web-GIS Applikation einfach und intuitiv erfolgen.

Die Vorgehensweise zunächst ist, die Geodaten, deren topologischen Beziehungen und Datenhaltung zu analysieren, um einen konsistenten Datensatz zu erzielen. Anschließend folgt eine Auseinandersetzung mit der ArcGIS Server Technologie und deren Frameworks und Bibliotheken zur Entwicklung eines Web-GIS Clients. Weiter werden die verwendeten Entwicklungswerkzeuge und Tools vorgestellt.

Am Ende des dritten Kapitels erfolgt eine Darstellung von zwei konzeptionellen Lösungsansätzen, wie mit der AGS Technologie topologische Beziehungen in einem Web-GIS Client integriert werden können.

3.1 GEODATEN IM LAFIS TESTSYSTEM

3.1.1 TESTDATEN

Die zugrunde liegenden Geodaten sind das Kernstück einer GIS Anwendung und sind deshalb genauer zu betrachten. Die Testdaten, die freundlicherweise das MLUV in Brandenburg zur Verfügung stellte, setzen sich aus folgenden Geodaten zusammen:

Geo-Basisdaten liegen zum Einem in Form von Vektordaten vor. Diese dienen hauptsächlich zur Orientierung für den Bearbeiter, um sich im Bearbeitungsgebiet zurechtzufinden. Dabei kommen Vektordaten von *Bundeslandgrenzen, Landkreisgrenzen, Gemarkungsgrenzen, Grenze des Fördergebietes und Blattschnitte der Topographischen Karte 1:25.000 (TK25)* zum Einsatz. Zum Anderen sind Luftbilder in Form von *Digitalen Orthophotos (DOP)* die Grundlage für die visuelle Erfassung von Antragsflächen. Diese Daten stammen von den Landesvermessungsämtern und werden von dem jeweiligen Landwirtschaftsministerium bereitgestellt.

Referenzdaten liegen als *Feldblöcke (engl. Physical Block, PB)* und *Landschaftselemente (LE)* im Vektorformat vor, deren Erzeugung jährlich aus den bearbeiteten Antragsdaten des

Vorjahres erfolgt. Diese Daten sind die Referenzfläche des Feldblockkatasters und damit gültige Basis für die Subventionsanträge der Landwirte.

Die Daten der *Feldblöcke* und *Landschaftselemente* liegen zusätzlich noch in *Bearbeitungsebenen* vor, ergänzt durch eine weitere Datenebene der *nicht förderfähigen Flächen (NEA, engl. Non Eligible Areas)*. In diesen drei Ebenen erfolgt die Bearbeitung der landwirtschaftlichen Förderflächen. Aus diesen drei Datensätzen erfolgt jährlich die Erstellung der Referenzflächen für das Folgejahr. Zusätzlich stehen in der Bearbeitungsebene *Daten aus der Ersterhebung* zur Verfügung, die auf Basis von Orthophotos digitalisiert wurden oder von GPS-Messungen stammen. Der Aufbau des ersten Feldblockkatasters fand hauptsächlich mit diesen Daten statt.

Während die Basisdaten und die Referenzdaten ausschließlich zur Visualisierung zum Einsatz kommen, soll die Bearbeitung der drei Feature Klassen (PB, LE und NEA) in der Bearbeitungsebene dynamisch über einen Webclient möglich sein.

3.1.2 Topologische Beziehungen der Testdaten

Für die zukünftige LaFIS-Web Anwendung sind im Vorfeld Produktspezifikationen festgelegt worden, die Konsistenzanforderungen und Topologiebeziehungen der editierbaren Testdatensätze enthalten. Diese sind im Folgenden zusammenfassend dargestellt.

1. Integritätsregeln (allgemeine Konsistenzbedingungen):
    a. Es sind nur Polygone für Referenzflächen zulässig
    b. Es sind keine Multi-Part Polygone erlaubt (wird ein Polygon geteilt, so müssen die beiden neuen Polygone jeweils eine neue eigenständige ID bekommen)
    c. Polygone haben eine Minimalfläche (PB: 10m²; LE und NEA: 1m²;)
    d. Die Genauigkeitsangabe der Koordinaten liegt bei 1mm
2. Topologieregeln
    a. Polygone dürfen sich nicht selbst überschneiden (self-intersections)
    b. Es sind keine übereinanderliegende Stützpunkte erlaubt (Mindestabstand 1cm)
    c. Referenzobjekte dürfen sich nicht überlappen (PB, LE, NEA)

d. Es sind nur Lücken zwischen den Referenzobjekten erlaubt, die größer als 10cm sind (kleinere Abstände gelten als SliverPolygons)
e. Aneinanderliegende Referenzobjekte teilen sich gemeinsam die Grenzlinie
f. Ein PB kann innerhalb eines anderen PB liegen
g. Ein PB kann NICHT innerhalb eines LE oder NEA liegen
h. Ein LE kann komplett innerhalb eines PB liegen, muss aber mindestens einen gemeinsamen Stützpunkt mit einem PB haben
i. LE können eine gemeinsame Grenzlinie mit eine anderen LE oder einer NEA haben
j. NEAs können innerhalb oder außerhalb von PB, LE oder anderen NEA liegen (keine spezielle topologische Regel für diese Feature Klasse)

Das Ziel dieser Arbeit ist es ‚Wege zu finden, die definierten topologischen Beziehungen in der zukünftigen LaFIS-Web Anwendung zu integrieren.

3.1.3 DATENHALTUNG IN ARCGIS UND ARCSDE

Eine Sammlung von geographischen Datensätzen in unterschiedlichen Formaten, die in Ordnern, Tabellen oder Datenbanksystemen vorliegt, wird in ArcGIS als eine Geodatabase bezeichnet. In der Geodatabase gibt es drei primäre Datensätze:

- Feature Klassen
- Raster Datensätze
- Tabellen

Letztere dienen zum Speichern von Attributdaten. (ArcGIS Desktop 1)

Feature Klassen sind im Prinzip auch Tabellen, die jedoch eine weitere Spalte mit Punkt-, Linien- oder Polygongeometrien enthalten. Raster Datensätze repräsentieren kontinuierliche räumliche Phänomene. (ArcGIS Desktop 1)

Erfolgt die Haltung der Geodaten in einem *relationalen Datenbankmanagement System (RDBMS)*, bietet ArcGIS eine Datenbankschnittstelle namens *ArcSDE* an, die auf verschiedenen RDBM-Systemen aufbauen kann und es ermöglicht, dort räumliche Daten zu speichern. Für die Speicherung dieser Daten wird von ArcSDE als Grundlage der OGC *Simple Feature Standard* verwendet. Die Erweiterung von ESRI eignet sich besonders bei großen Datenbeständen mit simultanen Zugriffen und versionierter Datenhaltung.

Für die Erstellung einer Topologie in ArcGIS müssen die in Beziehung stehenden Feature Klassen zu einem *Feature Dataset* zusammengefasst sein. Für dieses Feature Dataset und die darin enthaltenen Feature Klassen sind nun die topologischen Regeln zu definieren.

In den Tabellen des *ArcSDE Repository* werden die Eigenschaften der Topologie, die teilnehmenden Feature Klassen und die definierten topologischen Beziehungen gespeichert. Die Speicherung der topologischen Fehler erfolgt in Tabellen in Form von Error Objekten im Repository von ArcSDE.

Sind in einem RDBMS vorgehaltene Geodaten zu editieren, so empfiehlt es sich ein Versionierungskonzept zu erstellen und nur Versionen des Datensatzes zu editieren. Ist zudem eine Topologie für Geodaten festgelegt, so handelt es sich bei ihnen um keine Simple Features mehr. Dies hat auch zur Folge, dass das Editieren der Topologie nur in einer versionierten Geodatenbank möglich ist. Von den Versionen lassen sich Editierungen kontrolliert auf dem Ursprungsdatensatz (default Version) übertragen, dadurch ist die Vermeidung von Konflikten gegeben. Die Erstellung von Versionen ist in ArcGIS und ArcSDE möglich. Dazu muss der Datensatz zuerst als versioniert registriert und anschließend die Version erstellt werden. Dabei könnte ein Versionierungsszenario mit der Elternversion (default) und deren Kindversionen folgendermaßen aussehen:

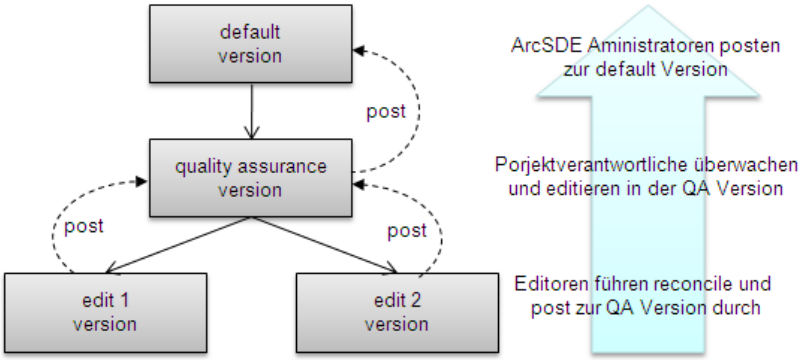


Abb. 3: Versionierungsszenario von Geodatenbanken [3]

Vor der Übertragung von Änderungen einer editierten Version in die Elternversion ist die Kindversion mit dem *reconcile* Befehl auf Konflikte zu prüfen. Dieser Vorgang vergleicht die Eltern- mit der Kindversion und zeigt bestehende Konflikte auf, die es zu bereinigen

[3] Quelle: (ArcGIS Desktop 2, 2009)

gilt. Danach erfolgt die Übertragung der Änderungen mit dem *post* Befehl auf die Elternversion. Die Kindversion steht dann für neue Editiervorgänge bereit.

3.2 DER ARCGIS SERVER

Der ArcGIS Server ist ein Produkt des Unternehmen ESRI Inc., kam im Mai 2004 mit der Version 9.0 auf den Markt und ist seit Juni 2008 in der Version 9.3 erhältlich (ArcGIS Server, 2009). Die Software ist in den drei verschiedenen Editionen Basic, Standard und Advanced verfügbar und weist je nach Edition einen unterschiedlichen Umfang an Funktionalitäten auf. Der ArcGIS Server bietet Möglichkeiten Karten, GIS-Werkzeuge und GIS Web Services im WWW zur Verfügung zu stellen. Zudem ermöglicht er durch die enthaltenen ArcSDE Schnittselle für Unternehmen eine zentrale Datenhaltung in RDBMS und die Einbindung von Geodaten aus unterschiedlichen Quellen aus dem Internet. Auf den ArcGIS Server kann mit verschiedenen Klienten zugegriffen werden, sowohl mit einer Desktop oder mobilen Anwendung, als auch mit einem Webbrowser mittels einer Web-Applikation. Ähnlich wie bei ArcGIS Desktop sind die Funktionalitäten durch Extensions erweiterbar. Der AGS ist in einer Java und .NET Version verfügbar. (ESRI D 1, 2009)

Zur Umsetzung von LaFIS-Web ist die Verwendung der Java Version des ArcGIS Servers 9.3 vorgesehen, unter anderem aus folgenden Gründen:

- ArcGIS Produktfamilie ist ein etabliertes System und bietet viele Bibliotheken zur Weiterentwicklung an.
- Das Java Web ADF zur Clienterstellung baut auf den W3C Standards auf. Dadurch ist die Integration von weiteren Frameworks möglich.
- Der ArcGIS Server ermöglicht nahezu den kompletten Funktionsumfang der ArcObjects für Web-GIS Clients.
- Es sollen in Zukunft bestehende LaFIS Produkte, die auf Java basieren, in die neue Web-Applikation integriert werden (z.B. GAF Massendruck Modul).

Der ArcGIS Server besitzt auch die Fähigkeit, Geodaten über die OGC Standards WMS, WFS und WFS-T zur Verfügung zu stellen. Somit könnte die LaFIS-Web Anwendung mit diesen Spezifikationen umgesetzt werden. In dieser Arbeit gilt es aber, die von ESRI bereit gestellten Konzepte und Methoden anzuwenden und mit diesen eine Web-GIS Anwendung

mit Editierfunktionen zu erstellen, sowie die Prüfung von topologischen Beziehungen zu integrieren.

3.2.1 Systemaufbau und Funktionsweise

Der *ArcGIS Server* ist ein verteiltes System und besteht im Wesentlichen aus den Komponenten Client, Webserver, GIS Server und Datenhaltung (siehe Abb. 4). Der folgende Abschnitt bietet darüber einen Überblick:

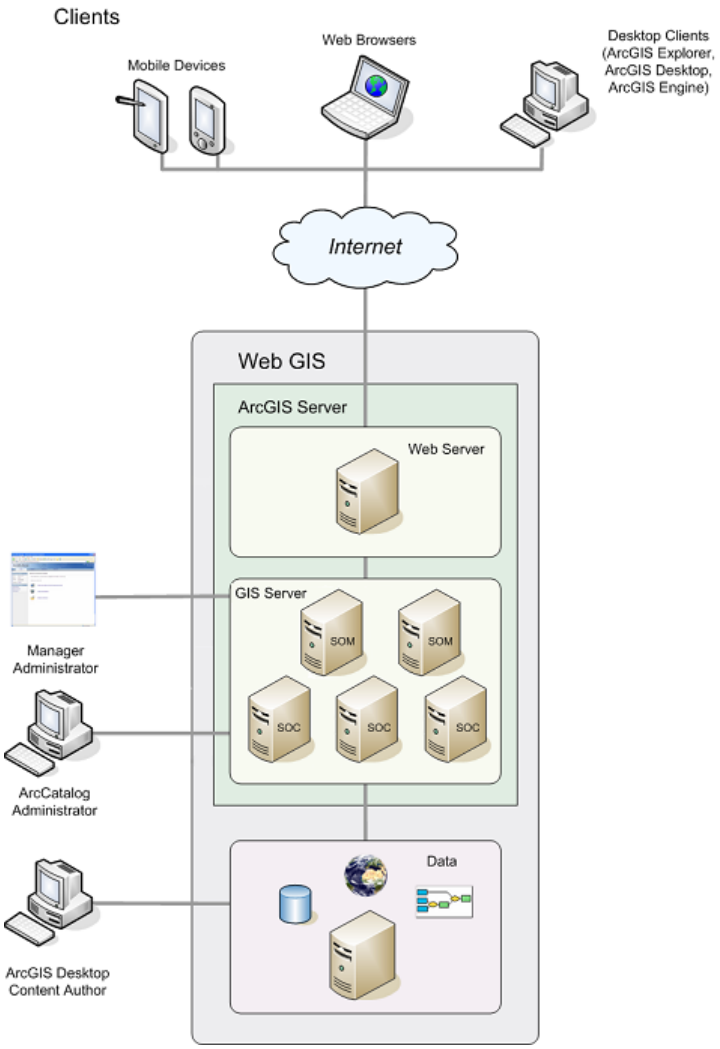


Abb. 4: Die AGS System Architektur [4]

Verschiedene *Clients*, wie *ArcGIS Desktop* Produkte oder *Webbrowser* können auf den ArcGIS Server zugreifen. Als Webbrowser können handelsübliche Produkte verwendet werden. Der AGS unterstützt dabei den *Microsoft Internet Explorer (IE)* in den Versionen

[4] Quelle: (AGS Help 1)

6.0, 7.0 und 8.0, sowie den *Mozilla Firefox* in den Versionen 2.0 und 3.0, die beide eine JavaScript Verarbeitung ermöglichen (ESRI Support Center 1). Die Web Clients stellen über den Browser eine Verbindung zu den im Webserver laufenden Web-Applikationen her. Desktop Clients verbinden sich mit den GIS Services über so genannte Web Service Endpoint URLs.

Der Webserver stellt *Web-Applikationen* und *Web Service Endpoints* bereit, welche mit den ADFs des AGS oder dem AGS-Manager erstellt wurden. Sie verwenden GIS-Ressourcen, die als Server Objekte in den *Service Object Containern (SOCs)* bereitstehen.

Der ArcGIS Server ist ein Teil der ArcGIS Produktreihe, die komplett auf ArcObjects Komponenten aufgebaut ist, die wiederum auf dem Component Object Modell (COM), einer Plattformtechnologie von Microsoft, basieren. So baut auch der ArcGIS Server auf den Software Komponenten der ArcObjects auf und wird daher von (ESRI Press, 2004) als ein Objekt Server für ArcObjects bezeichnet.

Der ArcGIS Server hält *GIS-Ressourcen*, wie z.B. Map Dokumente, vor und stellt diese als GIS-Services bzw. Server Objects im GIS-Server bereit. Der AGS besteht an sich aus dem *Server Object Manager (SOM)* und aus einem oder mehreren *Server Object Containers (SOCs)*. Der SOM ist ein Service im Betriebssystem des Server Rechners und fungiert als Verwalter der Server Objekte. Er nimmt die Anfragen an einen GIS-Service entgegen und leitet diese an die SOCs weiter, in denen die GIS-Services als Server Objects bereit gehalten werden. Die SOCs können je nach System Konfiguration und Server auf einen oder mehreren Rechnern verteilt sein. (AGS Help 2)

Die Administration des AGS erfolgt entweder über den *AGS-Manager* (Java und .NET Version, je nach Installation) oder über den *ArcCatalog*. Mit beiden Anwendungen ist es möglich, GIS-Ressourcen als Services auf dem AGS zu veröffentlichen. Der Abschnitt 3.2.2 stellt den ArcGIS Server Manager genauer dar.

Für die Erstellung von GIS-Services sind so genannte *GIS-Ressourcen* notwendig, die mit ArcGIS Desktop Anwendungen wie z.B. ArcMap oder ArcCatalog erstellt werden können. Einen Überblick über alle in AGS möglichen GIS-Ressourcen und den dazugehörigen GIS Services gibt folgende Tabelle:

GIS-Ressoruce	AGS Service	Funktionen
Map Dokument (.mxd, .pmf) .mxd mit einem tool layer .mxd das Daten aus einer versionierten Datenbank verwendet	Map Service	Mapping, Network Analysis, Mobile Data Access, WMS-, WFS-,WCS- und KML Publishing Geoprocessing Geodata extraction und replication
Address locator (.loc, .mxs SDE Batch Locator)	GeocodeService	Geocoding
Geodatabase (.sde, personal- oder file-GeoDB)	GeodataService	Geodatabase query, extraction und replication WCS publishing WFS publishing
Globe Dokument (.3dd, .pmf)	GlobeService	3D Mapping
Raster dataset Layer file der ein raster dataset referenziert compiled image service definitions (.ISCDef)	ImageService	Imaging WCS publishing
Toolbox (.tbx)	GeoprocessingService	Geoprocessing

Tab. 3: Übersicht der GIS-Ressourcen und AGS Services [5]

Wie zuvor beschrieben, erfolgt über die GIS-Ressource die Festlegung der Datenquelle. Im Fall des Map Services legen die im .mxd Dokument befindlichen Datenlayer den Zugang zu den Daten fest. Liegen die Daten in einer ArcSDE Geodatabase vor, erfolgt die Authentifizierung entweder über das Datenbanksystem oder über das Betriebssystem. Beim Ersteren ist für jeden Kartenlayer im Map Dokument der Username und das Passwort für den Zugriff auf die Datenbank gespeichert, über das der Map Service auf die Daten zugreifen kann.

Zur Erstellung einer Anwendung mit dem ArcGIS Server und zur Publikation von Geoinformationen über den AGS sind grundsätzlich folgende drei Schritte notwendig:

- *Aufbereiten* einer GIS-Ressource mittels ArcGIS Desktop Anwendungen. Die Art der GIS-Ressource ist abhängig von der GIS-Funktionalität, die im AGS veröffentlicht werden soll (siehe Tab. 3).
- *Veröffentlichen* der GIS-Ressource als GIS Service mittels AGS-Manager oder ArcCatalog. Dabei ist sicherzustellen, dass der SOC, in dem der Service

[5] Quelle: AGS Help 3

vorgehalten wird, Zugriff auf die verwendeten Dokumente und Datenressourcen hat.

- *Verwendung* des Services über eine Client Anwendung. Je nach Art des erstellten Services ist es möglich, diesen mit unterschiedlichen Clients zu verwenden.

(AGS Help 2)

Beispielsweise ist für eine Mapping Applikation ein .mxd Dokument in ArcMap als GIS-Ressource zu erstellen. Die Veröffentlichung des .mxd Dokuments als Map Service im ArcGIS Server erfolgt anschließend mit Hilfe des AGS-Managers. Der Map Service kann anschließend über eine, mit dem Web ADF erstellten Web-Applikation, verwendet werden.

3.2.2 DER AGS-MANAGER

Der AGS-Manager ist eine Browser basierte Anwendung zur Administration des ArcGIS Servers. Die Anwendung bietet Benutzeroberflächen für die Bewerkstelligung folgender Aufgaben (Auszug):

- Hinzufügen, konfigurieren, ändern und löschen von GIS Services und GIS-Ressourcen
- Erstellung von Web- und Enterprise Applikationen
- Hinzufügen von Host Maschinen (SOCs)
- Sichtung und Filterung von Log Dateien
- Vergabe von Nutzerrechten für Services

Der Manager ermöglicht GIS-Services in einer Ordnerstruktur anzulegen oder zu löschen und out-of-the-box mit Hilfe eines Wizards Webanwendungen für die laufenden GIS Services anzufertigen. Einstellungen und Eigenschaften der Services und Web-Anwendungen lassen sich ebenfalls über den Manager jederzeit ändern. Zudem bietet er Konfigurationsmöglichkeiten, die Services und Web-Anwendungen gegen unerlaubte Zugriffe zu sichern. Der Manager stellt außerdem eine Oberfläche zur Verfügung, in der die Log-Dateien der einzelnen Services gefiltert und gesichtet werden können. Die nächsten beiden Punkte erläutern die Einstellungen des Map Services und die der dazugehörigen Web-Anwendung.

3.2.3 DER AGS MAP SERVICE

Wie im vorherigen Punkt beschrieben, ist es möglich, auf einer GIS-Ressource aufbauend, verschiedene GIS Services anzulegen. So auch einen *Map Service*, der die Grundlage einer Mapping Applikation darstellt und deshalb im Folgenden näher betrachtet wird. Der Map Service ist der am häufigsten verwendete Service Typ im ArcGIS Server, der auf einem *ArcMap- (.mxd)* oder *published map file (.pmf)* Dokument basiert.

Vor dem Anlegen des Services ist jedoch die Erstellung eines Map Dokuments (GIS-Ressource) in ArcMap erforderlich. Folgende wichtige Aspekte sind dabei zu berücksichtigen:

- Statische Daten und sich sehr selten ändernde Daten sollten auf Grund besserer Performance *gecacht* werden. Dies entlastet den GIS Server und verbessert die Performance des Map Services.
- Die *Symbolisierung* dynamischer Daten sollte mit den optimierten Styles von ESRI erfolgen. Auf komplexe Symbolisierungen, wie z.B. Cartographic Lines, is nach Möglichkeit zu verzichten. Auch eine maßstabsabhängige Darstellung der einzelnen Ebenen, sowie die Verwendung von Annotation-Ebenen anstatt dynamisches Labeling wird empfohlen. Außerdem sollten alle Datenebenen im gleichen Koordinatensystem bzw. in der gleichen Projektion vorliegen.
- Das Map Dokument legt den Zugriff auf die Datenressource(n) fest. Für ein .mxd Dokument, das die Grundlage für eine Web-Editing Anwendung ist, müssen die Daten in einer ArcSDE Geodatabase vorgehalten werden. Die Zugriffsberechtigungen auf die Geodatenbank sind im Map Dokument gespeichert. Es muss jedoch sichergestellt sein, dass alle SOC Maschinen auf die .mxd Datei zugreifen können und die entsprechenden Berechtigungen haben.
- Die Darstellung der Layer Namen in der Web-Applikation ist identisch mit die der im .mxd Dokument.

(AGS Help 4)

Mit dem fertig gestellten Map Dokument ist die Einrichtung eines Map Services im ArcGIS Server möglich. Im AGS-Manager können für den Map Service, neben Name und GIS-Ressource, zusätzliche Einsatzmerkmale und Eigenschaften definiert werden. Die wichtigsten Einstellungen sind im Folgenden kurz dargestellt:

- Bei der Konfiguration der *Capabilities* ist es möglich festzulegen, ob der erzeugte Map Service als OGC Service (WMS, WFS oder WCS) oder KML Dienst verfügbar sein soll. Weiter besteht die Möglichkeit, den Dienst für mobile Endgeräte oder Netzwerkanalysen zu konfigurieren.
- Die Anzahl der zur Verfügung stehenden *Instanzen* ist für einen Map Services zu definieren. Eine vom Benutzer gestellte Anfrage wird über die Applikation zum SOM gereicht, dieser reicht die Anfrage an eine freie Service Instanz in den SOCs weiter. Aus Benutzersicht ist eine Service Instanz ein laufender Thread. Die Anzahl der möglichen Instanzen ist von der Leistung des Server Rechners bzw. von den SOCs, in denen die Service Prozesse laufen, abhängig. (ESRI Press, 2004)
- Das *Pooling* entscheidet über die Art der Verwendung der zur Verfügung stehenden Instanzen eines Map Services. Eine Instanz eines *Pooled Services* wird nach Verarbeitung einer Anfrage zurückgegeben und ist anschließend für eine andere Anfrage wieder verfügbar. Pooled Services sind nur für zustandslose (stateless) Operationen geeignet. Instanzen eines *Non-Pooled Services* hingegen werden einer einzigen Applikation zugeordnet und für die gesamte Dauer der Sitzung (session) verwendet. Diese Art von Service ist notwendig, wenn zustandsbehaftete (stateful) Operationen wie Undo/Redo zu verarbeiten sind. Non-Pooled Services laufen immer in einem eigenen Prozess des SOCs ab, sie sind also ein „high isolation" Service. (ESRI Press, 2004)
- Die Dauer der Verwendbarkeit einer Service Instanz kann durch die Vergabe von *Time-out* Zeiten geregelt werden.
- Handelt es sich um statische Daten, die nicht oder nur selten geändert werden, besteht die Möglichkeit den Map Service mit Hilfe des AGS-Manager zu *cachen*, um eine bessere Performance und kürzere Frage-/Antwortzeiten zu erzielen.

Nach der Erstellung des Map Services im ArcGIS Server ist er für Desktop Clients und Web-Anwendungen verfügbar.

3.2.4 WEB-GIS APPLIKATION „OUT OF THE BOX" MIT DEM AGS-MANAGER

Der ArcGIS Server Manager bietet die Möglichkeit, ohne Programmierkenntnisse eine lauffähige Web-Editing Applikation mit Hilfe eines Assistenten (Wizard) zu erstellen. Neben der Angabe eines Namens sind dazu im Wesentlichen fünf Schritte notwendig:

1. Festlegung des ArcGIS Server Typs (lokale oder Internet Verbindung)
2. Auswahl des GIS Services und der darzustellenden Layer,
3. Konfiguration der Tasks,
4. Hinzufügen von Kartenelementen und Auswahl des Layouts, sowie das
5. Deployen (Bereitstellung der Anwendung im Webserver für die Clients) der Anwendung.

(AGS Help 5)

Im ersten Schritt wird ein Service im ArcGIS Server über eine *lokale Verbindung (Local Connection)* oder über eine *Internet Verbindung (Internet Connection)* angesprochen. Dadurch entscheidet sich, welcher Verbindungstyp zwischen Webserver und GIS Server besteht und damit mit welchem Protokoll die Kommunikation zwischen den beiden Komponenten abläuft. Im Fall der Internet Verbindung ist dies *SOAP over HTTP*, womit aber nur die „grobkörnigen" ArcObjects im GIS Server mittels der SOAP API ansprechbar sind. Ist der Verbindungstyp lokal, so erfolgt die Kommunikation zwischen Web- und GIS Server über *DCOM/TCP*. Hierbei ist es möglich, im GIS Server mit den „feinkörnigen" ArcObjects zu arbeiten. Der Funktionsumfang ist mit diesem Verbindungstyp wesentlich größer und für Anwendungen mit Editierfunktionen notwendig. (ESRI Press, 2004)

Ist eine Verbindung zu einem AGS Map Service hergestellt, ist die *Auswahl der Layer* möglich, die der Service bereit stellt und in der Web-Anwendung gezeigt werden sollen. Die Layer können von einem oder mehreren verschiedenen Services (auch WMS, ArcWeb oder ArcIMS Services) in die Applikation eingebunden werden. Für jeden Layer ist es möglich die Attributfelder festzulegen, die für den Nutzer sichtbar sind.

GIS Funktionalitäten einer Web-Applikation können über so genannte *Tasks* definiert werden. Mit dem AGS Manager lassen sich vorgefertigte Tasks mit Hilfe eines Wizards erstellen und konfigurieren. Es stehen der *Editor*, *Find Address*, *Find Place*,

Geoprocessing, *Query Attributes*, *Search Attributes* und *Print Task* zur Verfügung. Der Punkt 3.2.5 beschreibt den Editor Task näher.

Nach der Definition der Tasks können im nächsten Schritt Kartenelemente der Web-Applikation hinzugefügt werden. Ein Element ist der *Table of Contents (TOC)*, der die sichtbaren Layer der Web-GIS Anwendung anzeigt. Zudem besteht die Möglichkeit, eine Übersichtskarte und einen Nordpfeil mit einzubinden. Ebenso kann für das Kartenfenster ein Copyright Vermerk angegeben werden. ESRI bietet für die Web-GIS Applikation eine Auswahl an verschiedenfarbigen Seitendesigns an, die auf unterschiedlichen CSS Dateien aufbauen.

Der letzte Schritt ist es, die erstellte Web-Applikation auf einem Webserver zu deployen und damit für die Anwender im Internet zur Verfügung zu stellen. Dafür bietet der AGS-Manager die Möglichkeit, die erstellte Anwendung mit zwei Mouseklicks auf dem internen Tomcat Webserver von AGS zu veröffentlichen. Dabei wird die Anwendung auf dem internen Tomcat Webserver des ArcGIS Servers gestellt, indem ein *Web ARchive File (.war)* in das WebApp Verzeichnis des Tomcat Webservers kopiert wird. Eine .war Datei ist eine gepackte Java Web-Applikation, die mit Hilfe eines Ant-Scriptes automatisch erstellt wurde. Ant von Apache ist ein in Java geschriebenes Tool zur automatischen Erzeugung von Programmen aus Quellcode.

Es ist jederzeit möglich, die Einstellungen zu den Layern, Tasks und Kartenelementen zu verändern. Danach ist nur ein erneutes Deployen der Anwendung notwendig. Der interne Webserver des AGS eignet sich jedoch nicht für den Produktionsbetrieb von Web-Anwendungen, sondern nur für Testzwecke. Für den Produktionsbetrieb wird dringend empfohlen, die Web-Applikation auf einem externen Webserver zu deployen, was auch bereits bei der Prototyperstellung umgesetzt wurde. Die generierte .war Datei lässt sich im AGS-Manager per Knopfdruck exportieren und anschließend für weitere Anpassungen und Entwicklungen in eine IDE (z.B. Eclipse) importieren.

3.2.5 Der Editor Task

Der Editor Task bietet eine Sammlung von Editierwerkzeugen, um über eine Web-Anwendung geographische Features in einer GeoDB zu editieren. Das Fenster mit den

einzelnen Werkzeugen (Commands and Tools) des Editier Tasks ist in der Abbildung 5 dargestellt.

Der Editor Task ist jedoch an bestimmte Bedingungen gekoppelt:

- Zwischen Web und GIS Server muss eine lokale (LAN) Verbindung bestehen. Für den Editor Task sind „feinkörnige“ ArcObjects notwendig (siehe 3.3.3).
- Die Feature Klassen müssen aus einer ArcSDE Datenbank sein, Shapefiles sind nicht editierbar.
- Die Editierung von Features in einer nicht versionierten DB kann mit einem pooled oder non-pooled Service erfolgen, die von Features in einer versionierten DB jedoch nur in einem non-pooled Service

(AGS Resource Center 1)

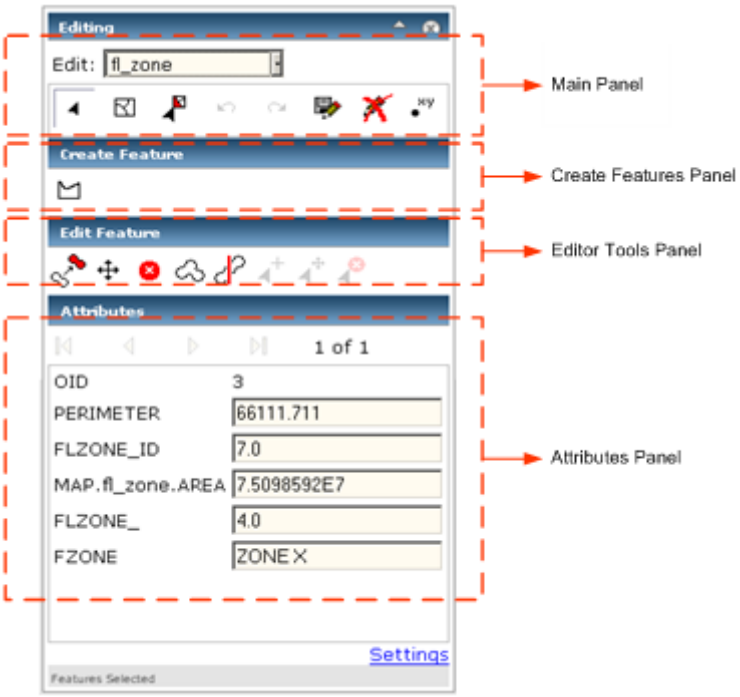


Abb. 5: Das Editor Task Fenster [6]

Die Konfiguration des Editor Tasks ist über den AGS-Manager möglich. Der Wizard zeigt nur die Layer und Datenbankversionen (falls vorhanden) an, die editierbar sind. Kommen die Daten aus einer versionierten Datenbank, so muss der angesprochene Map Service ein

[6] Quelle: (AGS Resource Center 1)

non-pooled Service sein. Treffen diese Konfigurationen nicht zu, erscheinen im Editor Task auch keine Layer, die editiert werden können.

Der Editor Task lässt sich darauf beschränken, dass nur das Editieren von Attributen, vorhandenen Features und ihrer Geometrie oder neuen Features möglich ist. Zudem können Snapping Toleranzen und Snapping Regeln, sowie die Farbgebung von markierten Features und Knoten konfiguriert werden.

Der Editor Task bietet bereits eine breite Palette an Editierwerkzeugen, deren Funktionalität jedoch vorgegeben ist. Der Task berücksichtigt z.B. bei der Bearbeitung keinerlei topologische Beziehungen, was in einer LPIS Anwendung jedoch zwingend erforderlich ist. Um weitere Funktionalitäten in den Editor Task einzubinden, besteht die Möglichkeit, diesen zu erweitern. Auf diese Weise kann einen neues Editier-Tool entwickelt werden, das topologischen Beziehungen berücksichtigt.

3.3 ARCGIS SERVER FRAMEWORKS UND APIS

Der ArcGIS Server steht, wie bereits erwähnt, in zwei Installationsvarianten zur Verfügung (.NET- und Java-). Dies hat u.a. Auswirkungen auf die Entwicklungswerkzeuge, die im Lieferumfang enthalten sind. Mit den SDKs der .NET Version sind Web-Applikationen mit ASP .NET Technologie und Mobile Anwendungen für ArcGIS Mobile möglich. Mit der Java Edition werden Frameworks für die Erstellung von Web- und Enterprise Anwendungen zur Verfügung gestellt *(AGS Web ADF und Enterprise ADF)*. Zudem kommt eine Bibliothek für die Erstellung von Web Services *(AGS SOAP API)* hinzu, sowie die *Java ArcObjects* Bibliothek. Das Web ADF und die letzten beiden Bibliotheken werden in den folgenden Punkten näher beschrieben.

3.3.1 AGS WEB ADF FOR JAVA

Mit dem Java *Web-Applikation Developer Framework (Web ADF)* können Webanwendungen erstellt werden, die geographischen Daten und GIS Funktionalität beinhalten und dabei allgemeine Web Standards wie HTML, CSS oder JavaScript verwenden. Das Java Web ADF ist mit JSF Komponenten nach dem *Model View Controller (MVC) Prinzip*, einem Entwurfsmuster für die Softwareentwicklung, aufgebaut und besteht aus drei Schichten (siehe Abb. 6). Das MVC Prinzip fordert eine strikte Trennung zwischen

der Präsentationsschicht und der Geschäftslogik, was den Entwicklungsprozess flexibler und wartbarer macht. (AGS Resource Center 2)

Die erste Schicht im Web ADF entspricht den *View/Controller* Einheiten von MVC und enthält die *Web Controls* des Web ADFs, die aus JSF Komponenten aufgebaut sind. Sie werden im Abschnitt 3.3.1.1 näher erklärt

Die *Model Schicht 1* enthält so genannte *Data Objects,* die direkt mit den Web Controls arbeiten und als Vermittler zwischen diesen und der Business Logik Klassen in der Model Schicht 2 agieren. Alle Data Objects werden vom *WebContext*, der zentralen Klasse einer Web-Anwendung, als Attribute verwaltet.

Die GIS-Ressource in der *Model Schicht 2* stellt den Zugang zu der/den *Datenquelle(n)* dar, die auch im WebContext registriert ist/sind. In der Model Schicht 2 sind zudem für jede GIS-Ressource die zur Verfügung stehenden GIS Funktionalitäten definiert, die wiederum mit Data Objects (Managed Beans) des WebContexts verbunden sind. Die Verwaltung und Konfiguration der Data Objects, GIS-Ressourcen und Funktionalitäten findet in der Datei faces-config.xml statt, die es für jede Web-Applikation gibt.

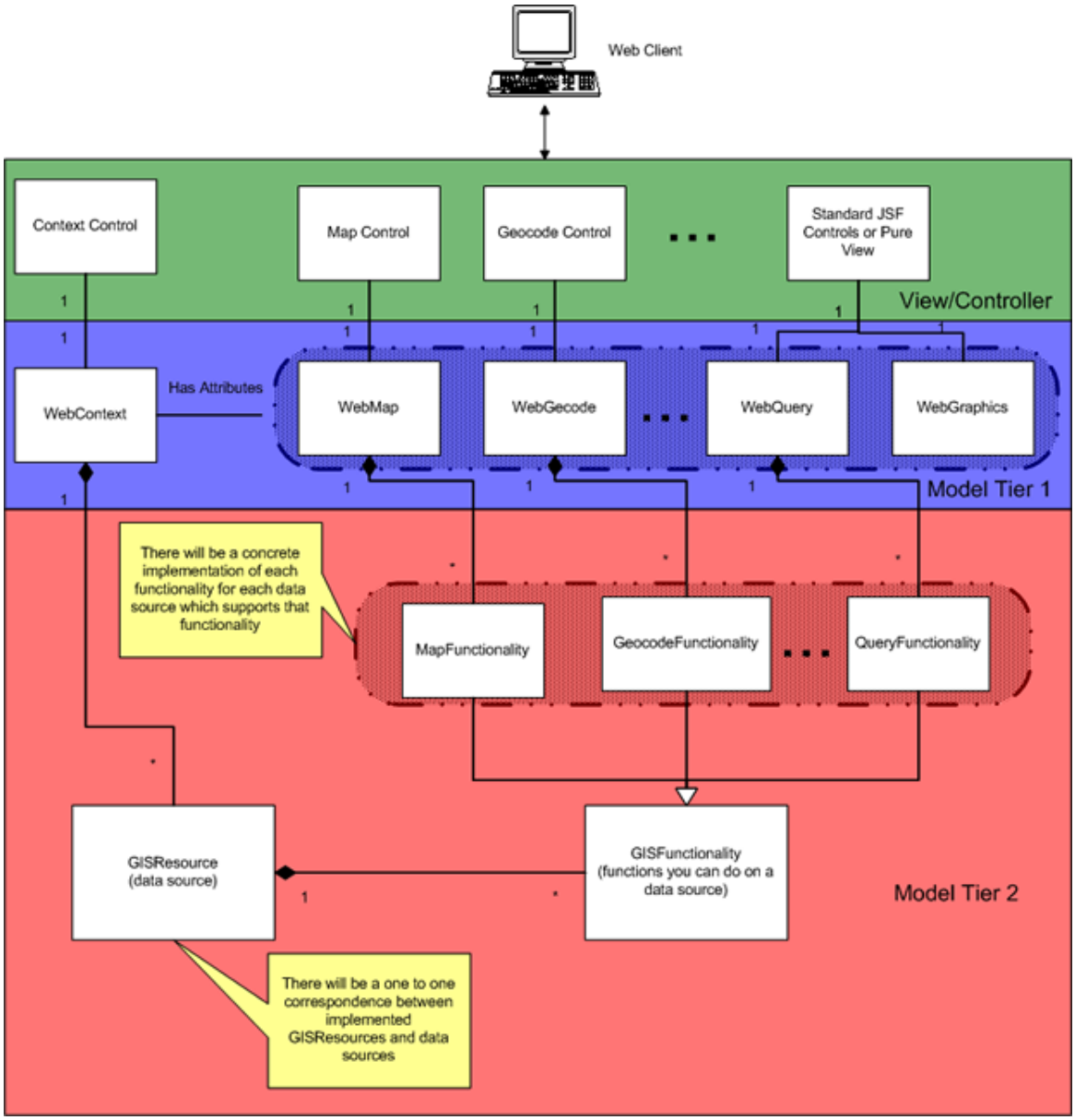


Abb. 6: Die AGS Web ADF Architektur [7]

3.3.1.1 Web ADF Controls

Ein Web ADF Controller ist ein Servlet, das mit einer Vielzahl von Java Klassen arbeitet und dabei die Interaktion zwischen der Modell- und View-Schicht kontrolliert. Die Web ADF Controls werden mittels der XML Datei *faces-config.xml* konfiguriert. Ein Control beinhaltet u.a. die Logik für den Seitenablauf in der Applikation und hat folgende weitere Funktionen:

[7] Quelle: (AGS Resource Center 3)

- Es bietet demn Benutzer eine visuelle Repräsentation einer Komponente (z.B. MapControl), mit dem der User interagieren und Events an den Controller schicken kann
- Der Controller delegiert die aufgerufenen Events an die Data Objects, welche die entsprechenden Model Klassen zur Abarbeitung der Aufgaben ansprechen.
- Der Controller kommuniziert mittels registrierter Listener mit den Model Klassen und nimmt die Antworten dieser entgegen.
- Der Controller stellt anschließend die Ergebnisse im View dar.

(ESRI Press, 2004)

Die Web Controls befinden sich in der View/Controller Schicht des Web ADFs und jedes Control arbeitet mit einem entsprechendem Data Object zusammen, das im WebContext als Attribut definiert ist. Data Objects sind *Managed Beans*. Managed deshalb, da sie vom JSF Framework über die Datei *faces-config.xml* verwaltet werden. Übernehmen Beans Werte aus einer HTML-Form, nennt man sie *Backing Beans*. Data Objects beinhalten nur wenig GIS Funktionalität, die meisten Aktionen werden an die Klassen der zweiten Datenschicht weitergegeben. Dadurch decken die Web Controls nicht die kompletten ArcObjects Funktionalitäten ab, bieten jedoch den Einstiegspunkt zu den ArcObjects im ArcGIS Server. (ESRI Press, 2004)

Das zentrale Control zur Steuerung der gesamten Web-Anwendung ist das *ContextControl*, das im Gegensatz zu den anderen Controls keine Visualisierungsfunktion hat. Es ist mit dem Data Object *WebContext* verbunden, der ein Container für alle Attribute und Ressourcen der Web-Anwendung ist und diese über Änderungen in der Applikation informiert. Das ContextControl baut über WebContext eine Verbindung zur GIS-Ressource auf und erhält diese aufrecht. In einer Web-Anwendung wird das ContextControl (= a:context) und die dazugehörige Managed Bean WebContext (=mapContext) in der JSP Haupseite (mapviewer.jsp) mit folgendem Tag eingebunden:

```
<a:context value="#{mapContext}" />
```

Listing 1: Das Context Control in mapviewer.jsp

In der Konfigurationsdatei faces-config.xml ist der WebContext als Managed Bean mit der zur Verfügung stehenden GIS-Ressource definiert. Mit der GIS-Ressource werden zudem

die dazugehörigen GIS Funktionalitäten definiert, die für die Ressource zu Verfügung stehen sollen.

Wie bereits erwähnt sind in faces-config.xml alle weiteren DataObjects im WebContext als Attribute registriert, so auch z.B. WebMap. Das MapControl ist für die Kartendarstellung in der Web-Applikation verantwortlich und ist über den WebContext an das Data Object WebMap gebunden:

```
<a:map value="#{mapContext.webMap}" id="map1" />
```

Listing 2: Das MapContext Control

Das MapControl stellt die Darstellungsfunktionalitäten für eine Karte, wie z.B. Pannen oder Zoomen, in einer Web-Applikation bereit und ist dabei AJAX unterstützt. An das MapControl sind über die mapId weitere Controls, wie das TocControl, OverviewControl oder ToolbarControl, gebunden, die in gleicher Weise wie das MapControl in die Haupseite mapviewer.jsp eingebunden werden. Beispiel TOC:

```
<a:toc id="toc1" value="#{mapContext.webToc}" mapId="map1" />
```

Listing 3: Das TOC Control

Im Web ADF stehen weitere Controls für Commands, Tools und Tasks zur Verfügung, die zur Ausführung von GIS Funktionalitäten notwendig sind. Die jeweiligen Controls übernehmen am Client Visualisierungsaufgaben und stehen zur Ausführung von GIS Operationen in Kommunikation mit den entsprechenden Business Klassen. Das Task Framework ist im nächsten Abschnitt näher erklärt.

3.3.1.2 Das Task Framework (GIS Werkzeuge im AGS Web ADF)

Tasks im ArcGIS Server Web ADF beinhalten Geschäftslogik mit deren Hilfe die Integration von GIS Funktionalitäten in eine Web-Anwendung erfolgt. Im Web ADF sind bereits mehrere Tasks vorhanden, wie z.B. der Map Tools Task oder der Search Attribute Task, die für alle GIS-Ressourcen verwendet werden können. Es gibt aber auch Ressourcen abhängige Tasks, wie z.B. den Edit Task, der nur für den ArcGIS Server

verwendbar ist. Tasks unterstützen AJAX und können *Parameter*, *Commands* oder *Tools* enthalten. (AGS Resource Center 4)

Commands werden von einem Client-Event (meist durch einen Button) ausgelöst und von einer serverseitigen Listener Klasse entgegengenommen und zur Business Logik delegiert. Ein Beispiel für ein Command ist „*zoom to full extend*“. Für das Command ist es möglich verschiedene Attribute zu definieren, wie z.B. Icons zur Darstellung in einer Toolbar.

Tools hingegen bestehen aus einer client- und anschließenden serverseitigen Aktion und erfordern eine Interaktion mit der Karte, wie beispielsweise das Aufziehen eines Rechtecks bei „*zoom in*“. Die clientseitigen Aktionen sind JavaScript Funktionen, die aus der *AGS JavaScript Bibliothek* zur Verfügung stehen. Die jeweiligen Aktionen und weiteren Attribute wie z.B. Images sind in der JSP Seite über das Tool-Control anzugeben.

Tasks sind einfache Java Klassen, die den Konventionen ähnlich der JavaBeans folgen. Zudem verwendet jeder Task eine eigene *TaskInfo* und *TaskConfig* Klasse, erstere enthält Metadaten wie Parameter-, Aktionen- und Toolbeschreibungen und letztere gibt den Zugang zu den Eigenschaften wie Beschriftung und Funktionalitäten. Mit dem *Task Framework* ist es möglich, neue Tasks zu entwickeln oder die vorhandenen Tasks zu erweitern, so auch den *Editor Task*.

Der Editor Task wurde bereits im Kapitel 3.2.5 vorgestellt. Im Folgenden sind die Komponenten aus dem Web ADF aufgelistet, die der Editor Task verwendet:

- **esri_edit.js:**
  JavaScript für die Client Aktionen im Browser
- **edit.jsp, mapvieser.jsp, settings.jsp, version.jsp und xy.jsp:**
  JSP Seiten für das Look & Feel des Editor Tasks
- **arcgis_webcontrols_ags.jar:**
  Web ADF Java Klassen für die Business Logik

(AGS Resource Center 1)

Der Editor Task und somit eine Instanz der EditBean wird bereits beim Start der Anwendung erstellt. Der Aufruf des Tasks in der Web-Anwendung löst eine JavaScript Funktion aus, die das Fenster mit den Editierwerkzeugen sichtbar macht. Steht mehr als eine Datenbankversion des Datensatzes zur Auswahl, die editierbar ist, so muss diese zuvor ausgewählt werden. Das Fenster ist aus mehreren JSP Seiten aufgebaut, die jeweils

Commands und Tools enthalten. Das Tool “Create Polygon” ist dabei folgendermaßen in die edit.jsp Seite eingebunden:

```
<a:button id="addPolygon" mapId="map1"
  clientAction="EsriEditingPolygon"
  serverAction="#{mapContext.attributes.mapEditor.addPolygon}"
  defaultImage="./images/tasks/editing/polygon.gif"
  hoverImage="./images/tasks/editing/polygonU.gif"
  selectedImage="./images/tasks/editing/polygonD.gif"
  toolTip="EditTask.TaskInfo.Tip.addpolygon" />
```

Listing 4: Das "add polygon" Tool in der edit.jsp Seite

a:button gibt das Web Control an, mit dem das Tool mit den zugehörigen Images in der Web-Anwendung dargestellt wird. Die *mapId* gibt die ID des Map Controls an und ist somit die Verbindung zur Map Ressource. Die *clientAction* gibt den Namen der JavaScript Funktion an, die das Zeichnen eines Polygons im Webbrowser ermöglicht. Bei *serverAction* ist der Verweis zur Edit Bean Klasse im Web ADF aufgeführt, die im WebContext (=mapContext) als Managed Bean konfiguriert ist.

3.3.2 AGS SOAP API

Ist bei der GIS-Ressource eine Internetverbindung definiert, so kann die Web-GIS Applikation auf den GIS Server nur mittels der SOAP API zugreifen. Die Kommunikation zwischen Web- und GIS Server erfolgt dabei mit *SOAP over HTTP*. Ist hingegen eine lokale (LAN) Verbindung zur GIS-Ressource festgelegt, so kann die Web-GIS Applikation entweder mit *SOAP over DCOM* oder mittels der ArcObjects mit *DCOM over TCP* mit dem GIS Server arbeiten. (AGS Resource Center 5)

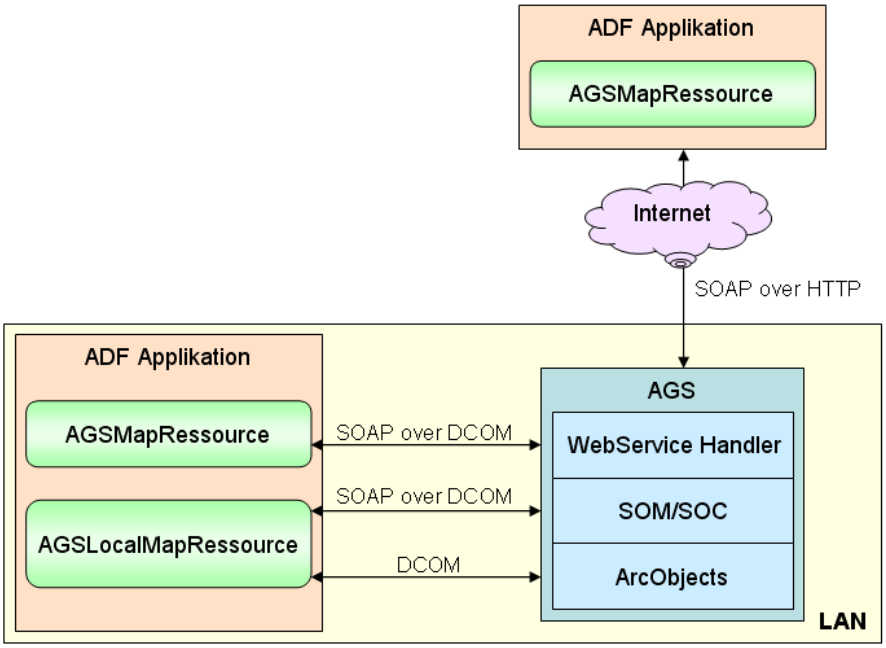


Abb. 7: Verbindungstypen zum ArcGIS Server

Einige Tasks verwenden nur Klassen aus der SOAP API, z.B. der „Query Attribute Task“. Für diesen Task reicht eine Internetverbindung aus. Dabei wird eine Verbindung über eine so genannte „Endpoint URL“ zu einem „grobkörnigen“ Server Objekt hergestellt, die nur ein Teil der ArcObjects Funktionalität wiedergeben. Diese Server Objekte werden im AGS internen Webserver als Web Services bereitgestellt. Im dazugehörigen (Web Service Description Language) WSDL File sind die so genannten „Value Objects“ und ausführbaren Methoden definiert, also die Funktionalität des Server Objektes. (AGS Resource Center 5)

Tasks die mehr GIS Funktionalität beinhalten (z.B. der Edit Task), Verwenden Klassen aus der ArcObjects API und erfordern deshalb eine lokale Verbindung. Somit ist es nicht möglich, Editierfunktionalitäten über die SOAP API abzubilden. Die ArcObjects API ist im nächsten Abschnitt näher beschrieben.

Vorteil der SOAP API ist, dass Value Objects lokal am Webserver erstellt und nur die Methodenaufrufe an den GIS Server gesendet werden müssen. Bei einer räumlichen Abfrage z.B. ist es somit möglich, den Spatial Filter mit seinen Attributen am Webserver zu definieren und nur für die räumliche Abfrage mit dem gesetzten Filter als Anfrage an den GIS Server zu schicken. Somit ist nur ein „Remote Call“ zum GIS Server notwendig. Bei einer Umsetzung mit den ArcObjects ist für jedes Setzen eines Attributwertes des Spatial Filters ein Remote Call zum GIS Server notwendig. Durch das Verwenden der

SOAP API reduziert sich somit der Datenverkehr zwischen Web und GIS Server. (AGS Resource Center 6)

3.3.3 ArcObjects Java API

Das Programmieren mit den *ArcObjects (AO)* im AGS gestaltet sich wie das Programmieren mit ArcObjects für Desktop Anwendungen. Der Unterschied ist u.a., dass es für jedes im AGS laufende „grobkörnige“ Server Objekt (z.B. MapServer) einen Stellvertreter (Proxy) im Web ADF gibt. Proxies im Web ADF erlauben den Zugriff auf die in den SOC Maschinen laufenden Server Objekte. Diese sind der Einstiegspunkt zu den „feinkörnigen“ ArcObjects in den SOC Maschinen des GIS Servers und somit zu den ArcObjects Funktionen und Methoden. (AGS Resource Center 7)

Im ArcGIS Server stehen u.a. folgende „grobkörnigen“ Server Objekte (Klassen) zur Verfügung:

MapServer, GeocodeServer, GeoDataServer, GeometryServer

Um mit einem Geometrieobjekt, das im Client (Browser) erzeugt worden ist, im GIS Server als AO Geometrie arbeiten zu können, sind folgende Transformationsschritte notwendig:

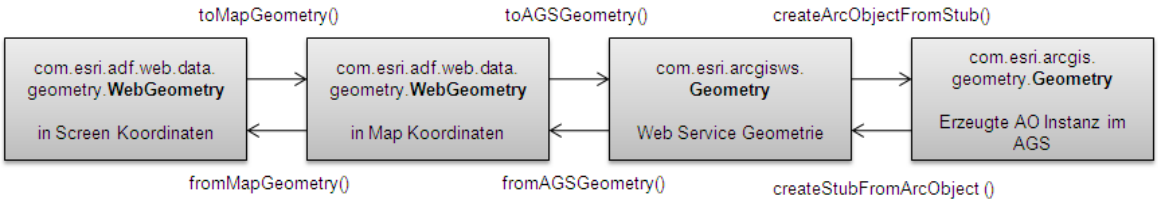


Abb. 8: Notwendige Transformationsschritte für Web-Geometrien

3.4 Werkzeuge, Tools und Methoden zur Weiterentwicklung

Zur Weiterentwicklung einer im AGS-Manager erstellten Web-GIS Anwendung sind weitere Entwicklungswerkzeuge und Methoden notwendig. Hierzu werden eine integrierte Entwicklungsumgebung (IDE - Integrated Development Environment), ein Build-Prozess-Tool und ein Test Webserver benötigt, die im nächsten Punkt vorgestellt werden. Zudem

finden bei der Weiterentwicklung Konzepte aus der objektorientierten Programmierung Verwendung, wie z.B. das Vererbungsprinzip.

3.4.1 VERWENDETE ENTWICKLUNGSWERKZEUGE UND TOOLS

3.4.1.1 Eclipse

Eclipse ist eine umfassende Entwicklungsumgebung, für die zahlreiche Open Source und kommerzielle Plug-ins existieren. Für die Weiterentwicklung von LaFIS Web kam die „Eclipse IDE for Java EE Developers“ zum Einsatz, mit der u.a. Enterprise- und Webanwendungen erstellt werden können. Diese Ausgabe enthält bereits das notwendige Plug-in „Web Developer Tools“, das wiederum die Bibliotheken der JSF Komponenten bereitstellt.

Auch ESRI bietet zum ArcGIS Server und zur ArcGIS Engine ein Plug-in für die Eclipse IDE an. Diese erleichtern die Entwicklung von ESRI Anwendungen, so kann z.B. ein AGS Web-GIS Projekt angelegt werden, das die GIS Server Verbindung und dessen Services gleich in die Anwendung mit einbindet. Zudem bietet das Plug-in verschiedene Vorlagen an, die als Grundlage für eine Web-GIS Anwendung dienen. Auf diese Weise entsteht ein Grundgerüst für ein Web-GIS Projekt, in dem die notwendigen ESRI Bibliotheken gleich eingebunden sind. Mit diesen Bibliotheken kann die Anwendung angepasst und erweitert werden. (AGS Resource Center 8)

3.4.1.2 Maven

Maven ist im Wesentlichen für zwei Aufgaben verantwortlich. Es verwaltet zentral die Programmbibliotheken für das Projekt und erledigt zusätzlich die Kompilierung/Übersetzung der Anwendung zu einem deploy-fähigen .war oder .ear File. Diese Datei wird in einem sogenannten *Build Process* zu einer *WAR (Web Archive)* Datei kompiliert und anschließend in einem Web- oder Application-Server veröffentlicht (deployed). Daraufhin kann sie in einem Webbrowser über eine URL aufgerufen werden. Für das Tool Maven steht in Eclipse ein Plug-in zur Verfügung.

Die Definition dieser Aufgaben erfolgt in einer zentralen Konfigurationsdatei, der *pom.xml (Project Object Model)* Datei. Das pom.xml File ist dabei in den *Dependency* Abschnitt für die Einbindung von Bibliotheken in das Projekt und in den *Deploy* Abschnitt für die

Konfiguration des Build Prozesses zur Erstellung der WAR Datei eingeteilt (Maven). Maven erwartet für den Build Prozess eine bestimmte Ordnerstruktur, die folgendermaßen aussieht:

```
my-app
|-- pom.xml
`-- src
    |-- main
    |   |-- java
    |   |   `-- SampleClass.java
    |   |-- resources
    |   |   |-- images
    |   |   |   `-- sampleimage.jpg
    |   |   `-- sampleresource
    |   `-- webapp
    |       |-- WEB-INF
    |       |   `-- web.xml
    |       |-- index.jsp
    |       `-- jsp
    |           `-- websource.jsp
    `-- test
    |   `-- java
    |       `-- SampleClass.java
    `-- target
    |    `-- webapp.war
```

Abb. 9: Projektstruktur für Maven, Quelle: (Maven), abgeändert

Diese Ordnerstruktur stimmt mit einer Java Web-Applikation und somit auch mit der im AGS-Manager erstellten Web-GIS Anwendung nicht komplett überein. Die Ordnerstruktur des Projektes muss leicht verändert werden, um mit Maven die GIS Anwendung verwalten zu können.

Ein Vorteil von Maven ist die zentrale Verwaltung der verwendeten Bibliotheken, die in der pom.xml Datei konfiguriert sind. So können neue Versionen von Bibliotheken, z.B. bei einer neuen ArcGIS Server Version, ganz einfach mit Änderung der Versionsnummer im pom.xml eingebunden werden. Alle Bibliotheken werden dabei zentral für alle Benutzer in einem so genannten *Repository* vorgehalten. Ein weiterer Vorteil ist die einfache Konfiguration des Build Prozesses, der mit einem kurzen Kommandozeilenbefehl aufrufbar ist. Die Konfigurationsdatei ist sogar so konfigurierbar, dass nach dem Build Prozess das erstellte .war File in das deploy-Verzeichnis des Webservers kopiert wird.

3.4.1.3 Apache Tomcat Webserver

Die dritte wichtige Komponente zur Weiterentwicklung im Testsystem ist der Webserver. Der ArcGIS Server beinhaltet zwar bereits einen Tomcat Web-Server, dieser eignet sich jedoch nicht für den Produktionsbetrieb einer Web-Applikation. Es ist daher empfehlenswert, die Anwendung von Beginn an mit einem eigenständigen Web-Server zu entwickeln, um spätere Komplikationen zu vermeiden. Der ArcGIS Server unterstützt den Apache Tomcat (Web) Server in der Version 5.5.17 und 6.0.13 und weitere Application Server wie Websphere, WebLogic und JBoss. Da der AGS den Tomcat als Web-Server selbst integriert hat und eine Unterstützung von Enterprise Java Beans (EJBs) für den LaFIS Web Prototypen nicht notwendig ist, werden für die Entwicklungsarbeiten der Apache Tomcat (Web) Server in der Version 6.0.13 verwendet.

Im Übrigen lässt sich der Tomcat Webserver in Eclipse einbinden. So ist es möglich, die Anwendung direkt im Eclipse internen Webbrowser oder den Webserver im Debug Modus zu starten, um in der Web-Anwendung zur Laufzeit nach Fehlern zu suchen.

3.4.2 PROGRAMMIERMETHODEN

3.4.2.1 Objektorientierung

Java ist eine *objektorientierte Programmiersprache*. Der Grundgedanke der Objektorientierung ist, dass Operationen und Zustände an Objekte bzw. Klassen gebunden sind. Konkrete Objekte einer Klasse werden als Instanzen bezeichnet. Klassen beschreiben somit die erzeugten Objekte, sie sind sozusagen deren Bauplan. Im Wesentlichen deklarieren sie Attribute und Operationen (Methoden). Die Instanzen haben jeweils eine Identität und einen Zustand und zeigen ein Verhalten. (Krüger, 2007)

3.4.2.2 Vererbung

Ein weiteres wichtiges Merkmal der Objektorientierung ist das *Vererbungsprinzip*. Dabei haben Unterklassen eine *Ist-eine-Art-von-Beziehung* zu ihren Oberklassen, z.B. eine Linie ist eine Art von Geometrieobjekt. Eine Klasse und eine Oberklasse haben sozusagen gewisse gemeinsame Merkmale. Beide Klassen haben gewissermaßen eine hierarchische Verbindung, es geben die Oberklassen den Unterklassen gewisse Eigenschaften mit.

In Java erweitert eine Unterklasse eine Oberklasse durch das Schlüsselwort *extends*. Eine Oberklasse vererbt alle sichtbaren (nicht private) Eigenschaften und Methoden an die Unterklasse. In Java ist nur eine Einfachvererbung möglich, wodurch Probleme vermieden werden, die durch Mehrfachvererbung auftreten können. Die Mehrfachvererbung, bzw. die Annahme von unterschiedlichen Typen, ist in Java durch die Implementierung mehrerer Schnittstellen umsetzbar. (Krüger, 2007)

3.4.2.3 Schnittstellen

Eine Schnittstelle (engl. Interface) enthält ausschließlich abstrakte Methoden und Konstanten und wird von Klassen implementiert. Interfaces legen somit die Methoden in den Klassen fest, von der sie implementiert werden. Dabei kann eine Klasse eine oder mehrere Schnittstellen mit dem *implements* Schlüsselwort implementieren. (Krüger, 2007)

3.4.2.4 Methoden überschreiben

Ist eine Methode einer Oberklasse mit demselben Methodennamen und gleicher Signatur in einer Unterklasse deklariert, so wird diese überschrieben. Die Methode in der Unterklasse kann sozusagen die Methode aus der Oberklasse spezialisieren, indem sie Eigenschaften aus der Unterklasse verwendet. (Krüger, 2007)

3.5 KONZEPTIONELLE LÖSUNGSANSÄTZE

Es gibt verschiedene Ansätze Geodaten nach topologischen Regeln und räumlichen Beziehungen in einer Web-GIS Applikation zu prüfen. Für diese Arbeit wurden folgende zwei Herangehensweisen näher betrachtet. Zum Einen gibt es die Möglichkeit, Geodatensätze wie in ArcGIS bzw. ArcMap zu prüfen, indem auf der Datenbankseite topologische Regeln für die Datensätze definiert werden und anschließend die Topologie für ein bestimmtes Gebiet validiert wird. Zum Anderen enthält die ArcObjects Bibliothek räumliche Relationsparameter und topologische Operatoren für die Prüfungen und Bereinigung einzelner Geometrieelemente.

Der Hauptunterschied ist, dass bei erst genannter Methode ein Datensatz geprüft wird, der beispielsweise mehrere Änderungen eines ganzen Tages beinhaltet. Mit den räumlichen Relationsparametern und topologischen Operatoren hingegen können einzelne

Geometrieobjekte mit anderen in Beziehung gebracht und z.B. während eines Digitalisiervorgangs topologische Beziehungen überprüft werden.

3.5.1 Topologie in der Geodatabase

Dieser Ansatz beruht auf der Verwendung der topologischen Prüfungsmechanismen in der Geodatabase. Als Voraussetzung wird angenommen, dass die editierbaren Layer in einem Feature Dataset zusammengefasst sind, zu dem die topologischen Beziehungen (siehe 3.1.2) definiert wurden. Diese lassen sich mittels der *validate* Funktion überprüfen und die resultierenden Ergebnisse in einem Topologie Layer darstellen. Der entstandene Topologie Layer könnte z.B. in die Web-Applikation integriert und zur Anzeige der Topologiefehler verwendet werden. Zur Bereinigung der Fehler ist allerdings die Neuentwicklung von Editiertools zur Bearbeitung der Topologie notwendig, wie sie z.B. in der *Topology* Werkzeugleiste in ArcMap zu finden sind. Weiter ist noch ein Dialog für die „reconcile“ und „post“ Funktionen von versionierten Datenbanken zu erstellen. Folgendes Ablaufdiagramm zeigt für diese Umsetzung zusammenfassend die dazu notwendigen Schritte:

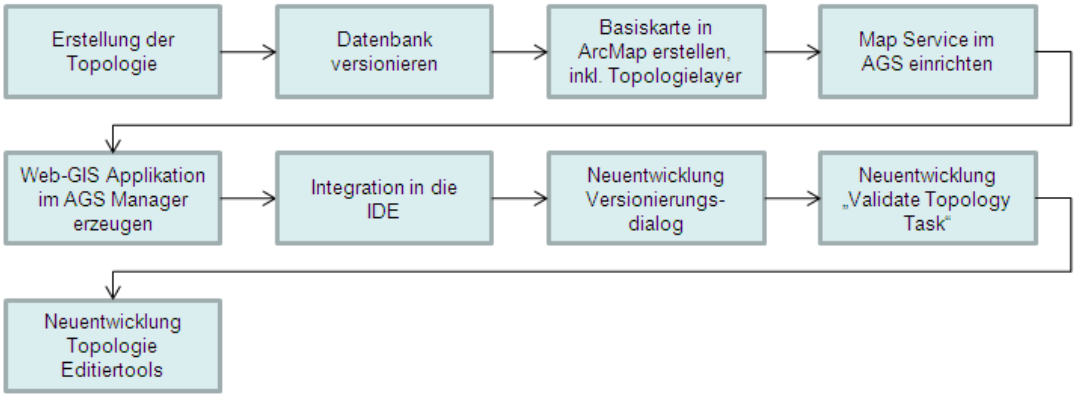


Abb. 10: Arbeitsschritte für die Geodatabase Variante

Die in der Abbildung dargestellten Neuentwicklungen für die Web-GIS Applikation müssen dabei folgende Funktionalitäten enthalten:

- **Versionierungsdialog**

  Funktionalitäten für den „reconcile“ und „post“ Prozess sind notwendig, um die Änderungen zu der Elternversion zu übertragen zu können.

- **Validate Topology Task**

  Bevor eine Editierversion zu der QA-Version gepostet werden kann, ist die Prüfung der Topologie durchzuführen. Der *Validate Topology Task* stößt den Validierungsprozess der Topologie in der Datenbank an und überprüft dabei wahlweise den aktuellen Kartenausschnitt (current extent) oder die Gebiete wo Editierungen vorgenommen wurden (dirty areas). Topologiefehler sind anschließend im *Topology Layer* zu sehen. Der Task soll außerdem ein Tool (Fix Topology Error Tool) enthalten, mit dem der User interaktiv die topologischen Fehler auswählen und korrigieren kann.
- **Topologie Editier Tool**

  Für das Auswählen und Editieren von topologischen Elementen ist außerdem ein Tool nötig. Dieses Werkzeug könnte zum Beispiel mit in den Editor Task integriert werden.

Die Umsetzung dieser Variante erfordert einen großen Arbeitsaufwand und tiefgreifende Programmierkenntnisse. Hierzu sind Anpassungen sowohl im Client Bereich (JavaScript und Web ADF) als auch umfangreiches Programmieren mit den ArcObjects des GIS Server notwendig. Zudem könnte für die Endanwender das Arbeiten mit der versionierten DB, der Topologie Prüfung und das Bereinigen der Topologiefehler mit den Tools zu kompliziert werden, da sie keine tiefgreifenden GIS Kenntnisse besitzen. Alle in 3.1.2 festgelegten topologischen Beziehungen sind jedoch nicht über die Topologie der Geodatabase überprüfbar, weshalb z.B. die Prüfung der Minimalfläche (Integritätsregel 1d) anderweitig implementiert werden müsste.

Die Entwicklung dieses Konzeptes hat aber auch einen wesentlichen Vorteil. So wäre dieser Entwicklungsprozess nur einmalig notwendig und könnte für alle weiteren Web-GIS Anwendungen eingesetzt werden.

3.5.2 Erweiterung des Editor Tasks

Ein zweiter Lösungsansatz verfolgt die Erweiterung der Editierfunktionalitäten und somit eine Erweiterung des Editor Tasks im AGS Web ADF. Dabei werden in die bestehenden Editierwerkzeuge Funktionalitäten zur Überprüfung und Bereinigung der Topologie integriert, so dass ein konsistenter Datenbestand erzielt wird. Es ist möglich die topologischen Beziehungen in Kombination mit räumlichen Abfragen und topologischen

Operatoren zu überprüfen. Bei diesem Ansatz erfolgt die Kontrolle für jedes Geometrieelement während des Editierens. Für diese Umsetzung sind folgende Arbeitsschritte notwendig:

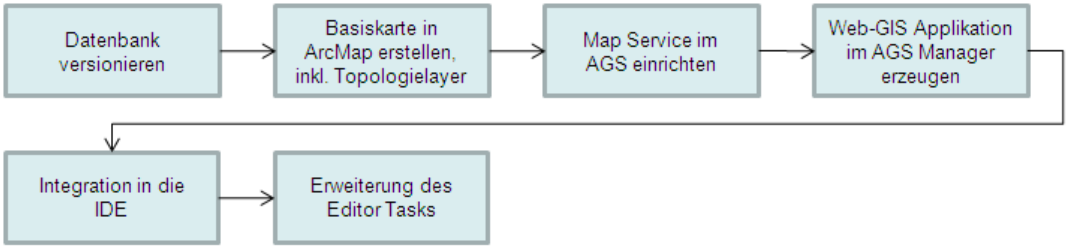


Abb. 11: Arbeitsschritte für die Erweiterung des Editor Tasks Variante

Der Editor Task soll so erweitert werden, dass ein Geometrieelement während des Editiervorganges unter Verwendung des ArcGIS Servers auf topologische Fehler geprüft und auch gleichzeitig bereinigt wird. Am Ende soll der User ein topologisch korrektes Geometrieelement in der Web-GIS Anwendung erhalten, das in der DB gespeichert wird. Für dieses Anwendungsszenario wurde folgender System Use Case erstellt.

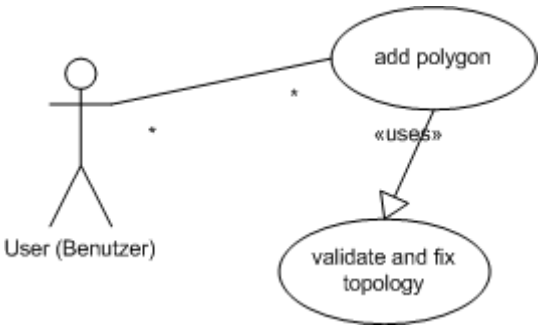


Abb. 12: Anwendungsfall "add Polygon"

Beschreibender Systemanwendungsfall	
Name	**Add polygon**
Kurzbeschreibung	Der User editiert im Kartenfenster der Web-GIS Anwendung ein neues Polygon
Akteure	Client (Web-GIS Anwender)
Auslöser	Der User klickt auf den Button „create polygon“
Ergebnis(se)	Ein topologisch korrektes Polygon wird der Web-GIS Anwendung hinzugefügt

Eingehende Daten	Die vom User gezeichnete Bildschirmgeometrie
Vorbedingungen	Der User hat den Editor Task geöffnet und den zu editierenden Layer ausgewählt.
Nachbedingungen	Es können weitere Polygone editiert werden.
Essenzielle Schritte	- Polygon im Browser zeichnen - Polygon zum Webserver schicken - Include: Validate and fix topology - Polygon der Web-GIS Anwendung hinzufügen
Offene Punkte	-
Änderungshistorie	Erstellt am 09.08.2009
Sonstiges, Anmerkungen	-

Tab. 4: Anwendungsfall "Add polygon"

Beschreibender Systemanwendungsfall	
Name	**Validate and fix topology**
Kurzbeschreibung	Die eingehende (gezeichnete) Geometrie wird auf definierte topologische Beziehungen geprüft. Dabei erfolgen die Bereinigung von topologischen Fehlern und die Rückgabe eines topologisch korrekten Polygons.
Akteure	Webserver
Auslöser	Ein Polygon wurde gezeichnet
Ergebnis(se)	Ein topologisch korrektes Polygon
Eingehende Daten	Geometrieobjekt aus dem Map Event
Vorbedingungen	-
Nachbedingungen	-
Essenzielle Schritte	- Web Geometrie in GIS Geometrie umwandeln - Räumliche Abfrage der in Beziehung stehenden Geometrieelemente durchführen - Prüfung auf topologische Fehler - Bereinigung der topologischen Fehler
Offene Punkte	
Änderungshistorie	Erstellt am 09.08.2009
Sonstiges, Anmerkungen	

Tab. 5: Anwendungsfall "Validate and fix topology"

Diese Lösung ist für den Endnutzer sehr einfach zu bedienen. Er benötigt keine weiteren Kenntnisse über die Topologie in einem Datensatz und deren Fehlerbehebung. Zudem

kann ein toplogisch korrekter Datenbestand in nur einem Arbeitsschritt, dem Editieren, erreicht werden. Für das Prüfen der Topologie und Beseitigung ihrer Fehler sind keine separaten Arbeiten notwendig. Zudem hält sich der Implementierungsaufwand für diesen Ansatz in Grenzen und kann schnell umgesetzt werden.

Jedoch erfolgt die Beseitigung der topologischen Fehler in diesem Ansatz nach fest definierten Regeln. Der Endnutzer hat keinen Einfluss darauf, wie die topologischen Fehler bereinigt werden. Zudem könnten sich die zusätzlichen Prüfungen und Operationen schlecht auf die Performance und somit auf die Nutzerakzeptanz auswirken. Ein weiterer Nachteil ist die spezielle Implementierung für jede Web-GIS Anwendung, lediglich die entwickelten Klassen und Funktionen können wiederverwendet werden.

Für den Web-GIS Client besteht allerdings die Anforderung, dass er für den Endnutzer einfach und intuitiv zu bedienen ist. Aufgrund der leichten Handhabung für den Endnutzer und der schnellen Implementierung wurde für die Umsetzung daher dieser Lösungsansatz ausgewählt.

4 UMSETZUNG

Es wird davon ausgegangen, dass sich die Geodaten bereits in einer relationalen Datenbank (Oracle 11g) befinden und ArcSDE eingerichtet ist. Die Datenbankverbindung ist über das LAN erreichbar. Die Umsetzung des in 3.5.2 beschriebenen Lösungsansatzes teilt sich demnach im Wesentlichen in folgende Schritte auf:

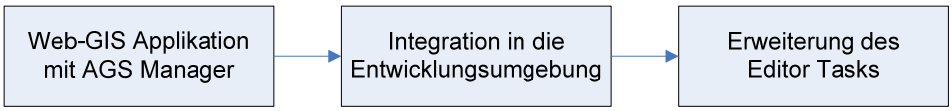


Abb. 13: Überblick der Arbeitsschritte für die Umsetzung

4.1 WEB-GIS APPLIKATION MIT DEM AGS-MANAGER

Für die Erstellung einer Web-Mapping Anwendung muss, wie in 3.2.1 dargestellt, eine GIS-Ressource mit ArcMap und darauf aufbauend ein Map Service im ArcGIS Server erstellt werden. Anschließend ist mit dem AGS-Manager die Erstellung einer Web-GIS Applikation mit Editierfunktionen möglich.

4.1.1 BASISKARTE IN ARCMAP

Als GIS-Ressource für den AGS Map Service dient ein mit ArcMap erstelltes .mxd Dokument. Dabei wurde in ArcCatalog eine ArcSDE Datenbankverbindung zu den LaFIS Daten eingerichtet und hergestellt. Aus der DB erfolgte der Import der in 3.2.1 vorgestellten Datensätze mit anschließender Anordnung in der richtigen Reihenfolge.

Die Symbolisierung der einzelnen Layer erfolgte in Anlehnung an die LaFIS Symbole und ausschließlich mit den ESRI Optimized Styles, um eine schlechte Performance wegen zu langer Renderzeiten zu vermeiden. Folgende Abbildung zeigt die Legende (ToC) des erstellten Map Dokuments in ArcMap.

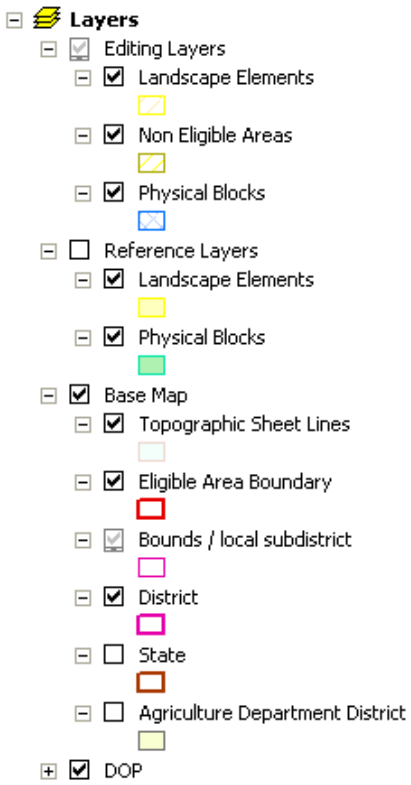


Abb. 14: Ebenenstruktur und Symbolisierung in ArcMap

4.1.2 AGS Map Service für die Web-GIS Anwendungen

Auf Basis der im vorherigen Abschnitt erstellten .mxd Datei ist im ArcGIS Server ein Map Service mit dem Namen „MapService_lafis_web“ eingerichtet worden. Die Konfiguration für folgende Einstellungen fiel auf folgende Parameter:

Capabilities: Die LaFIS-Web Anwendung wird vorerst ausschließlich mit dem erstellten *AGS Map Service* betrieben. Ein anderer Service, z.B. WMS, ist nicht vorgesehen, kann aber nachträglich jederzeit und ohne Aufwand konfiguriert werden.

Pooling: Ein *not pooled* Service ist für eine versionierte Datenbank notwendig. Ein Vorteil des *not pooled* Services ist, dass einzelne Editierschritte wieder rückgängig gemacht werden können.

Instanzen: Es wurden sechs Instanzen angelegt, somit können sechs Benutzer die Web-GIS Anwendung gleichzeitig öffnen. Dies ist für die Testimplementierung vorerst ausreichend. In den *System Design Strageties* von ESRI werden max. 3 – 4 Instanzen pro CPU Core empfohlen, so dass je nach Serverleistung nur eine begrenzte Anzahl an Usern arbeiten können. Für höhere Userzahlen müssen mehrere SOC Maschinen im System

installiert und bei sehr hohen Userzahlen sogar zwei SOM Server mit *Load Balancing* zur gleichmäßigen Arbeitsverteilung eingerichtet werden.

Ein *Cachen der statischen Daten* fand für die Testanwendung nicht statt. Für ein Produktivsystem empfiehlt es sich jedoch, nicht veränderbare Daten, wie beispielsweise administrative Grenzen, für eine bessere Performance zu cachen. Dadurch verringeren sich die Renderzeiten der Karten und die Webapplikation ist so wesentlich performanter.

Der eingerichtete Map Service ist unter *Services* > *Manage Services* > *Maps* aufgelistet (siehe Abb. 15) und kann zu jeder Zeit geändert werden. Ist das zu Grunde liegende .mxd Dokument verändert worden, so ist ein Neustart des Services notwendig.

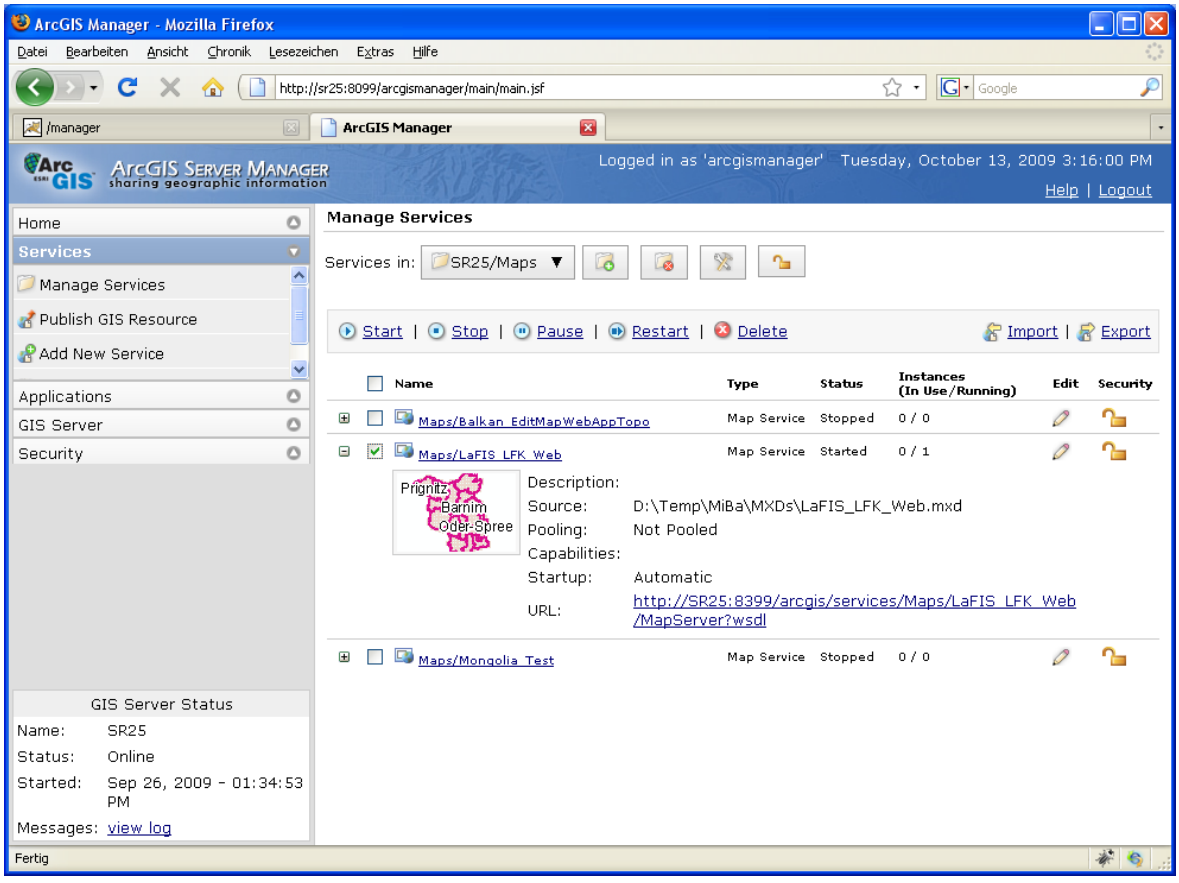


Abb. 15: Der Map Service für LaFIS-LFK im AGS-Manager

4.1.3 Die Web-Applikation mit dem AGS-Manager

Im AGS-Manager ist es möglich, mit einem Wizard in mehreren Schritten eine voll funktionsfähige Web-Mapping Anwendung mit Editierfunktionen zu erstellen (siehe 3.2.4 und 3.2.5). Die erzeugte .war Datei und somit auch der Quellcode der Applikation lässt

sich aus dem AGS-Manager exportieren und anschließend in eine Entwicklungsumgebung importieren. Die automatisch generierte Anwendung und ihr Quellcode stehen somit als Ausgangsbasis für die Weiterentwicklung zur Verfügung.

Dieser Abschnitt stellt im Folgenden kurz die wichtigsten Einstellungen und Konfigurationen für die LaFIS-Web-Applikation dar, die im AGS-Manager Wizard zur Erstellung einer Web-Anwendung vorgenommen wurden.

Für LaFIS Web wurde eine *Verbindung* zu einem *lokalen ArcGIS Server* ausgewählt, d.h. der Web- und GIS-Server befindet sich im gleichen LAN. Diese ermöglicht das Arbeiten mit den so genannten *„fine grained“* ArcObjects im GIS Server, was für Editing Anwendungen unabdingbar ist.

Ist die Verbindungsart festgelegt, so kann der zuvor angelegte Map Service als *GIS-Ressource* ausgewählt und die darin enthaltenen Layer der Webanwendung hinzugefügt werden. Nach Bedarf lässt sich die Sichtbarkeit der Attribute einschränken und die Transparenz der Layer anpassen.

Auf Basis der hinzugefügten Karteninhalte werden im nächsten Schritt die *Tasks* konfiguriert. Die LaFIS-Web Applikation enthält den *„Search Attribute“* und *„Editor“ Task*. Für den erst genannten Task sind die Attributfelder der Feature Klassen festgelegt worden, deren Einträge über die Suchmaske auffindbar sind. Die Suchmaske enthält nur ein Eingabefeld, das gesuchte Wort wird in allen festgelegten Attributfeldern gesucht. In der LaFIS-Web Anwendung ist die Suche nach Feldblocknummern, Gemarkungen, Gemarkungsnummern und Landkreisen möglich.

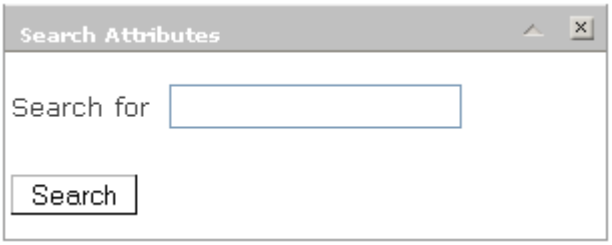


Abb. 16: Das "Search Attribute" User Interface in LaFIS-Web

Der zweite konfigurierte Task, der *Editor Task*, ist stark von der vorliegenden GIS-Ressource und der darin enthaltenen Layer abhängig (siehe 3.2.5). Bei der Konfiguration des Tasks sind im *General Tab* nur die Layer des Map Services sichtbar, die die Anforderungen für editierbare Feature Klassen erfüllen. Lediglich die Layer der

Bearbeitungsebene (PB, LE und NEA) sind ausgewählt worden, alle anderen Datensätze sind nicht editierbar. Es sind für die Testanwendung zwei Versionen der Daten definiert worden, somit erfolgt das Editieren entweder in der *Edit_A* oder *Edit_B* Version.

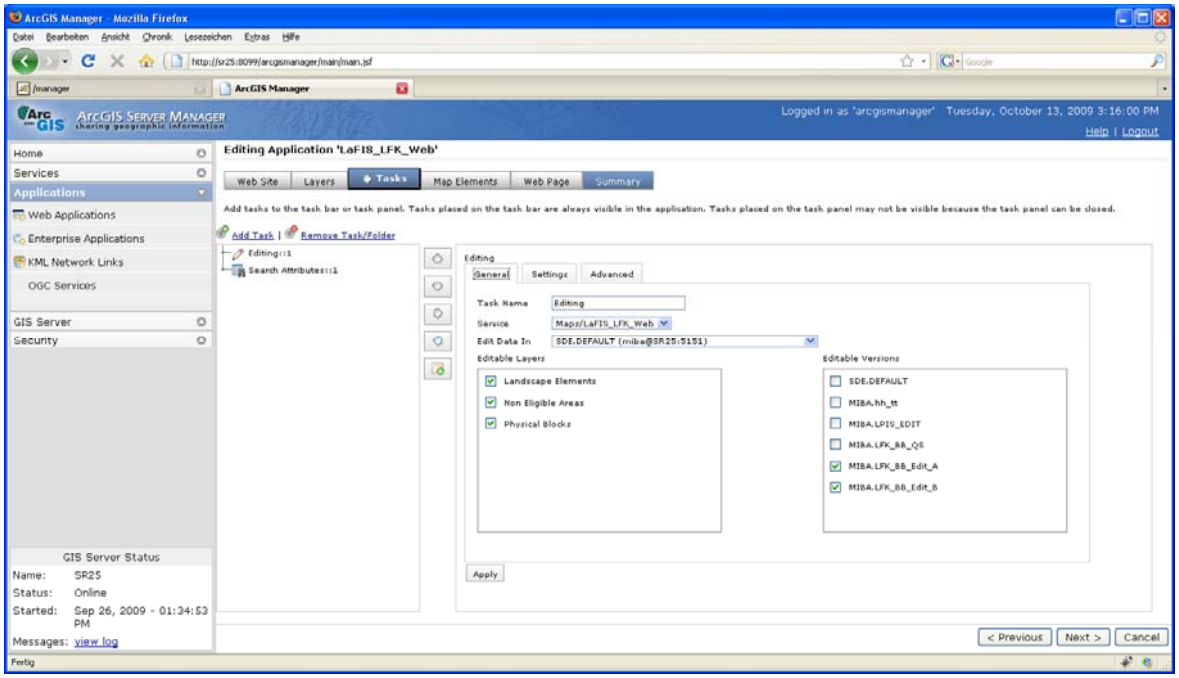


Abb. 17: Konfiguration des Editor Tasks im AGS-Manager

Im *Settings Tab* erfolgte die Festlegung des vollen Editierumfangs für die drei Feature Klassen, d.h. es ist möglich neue Features zu erstellen, bestehende Features zu verändern und deren Attribute zu editieren. Zusätzlich ist eine Selektions- und Snapping-Toleranz von acht Pixeln definiert worden, damit ein Feature nicht pixelgenau ausgewählt bzw. ein zu editierendes Feature an bestehende Objekte „gesnappt“ wird. Die Snapping-Funktion wurde für Edges, Vertices und Ends festgelegt. Sie soll den User das Editieren erleichtern um einfacher neue Objekte an bestehende Features ohne Überlappungen oder Lücken angrenzen zu lassen.

Editieren mehrere User gleichzeitig einen Datensatz, sind Konflikte möglich. Der *Advanced Tab* bietet Einstellungsmöglichkeiten im Umgang mit Konfliktsituationen. Es wurde festgelegt, dass Änderungen bei gleichzeitiger Bearbeitung der Attribute eines Objektes nur die von einem User gelten („Row Level“) und nur die jeweils aktuellste Änderung zu verwenden ist („Child Win“).

Nach Konfiguration der beiden Tasks sind die Auswahl des *Website-Layouts (sog. Theme)* und das Hinzufügen von verschiedenen *Kartenelementen* möglich. Für LaFIS-Web fiel die

Wahl auf das Standard Layout, welches zu einem späteren Zeitpunkt noch an das LaFIS Layout angepasst wird. Zudem enthält die Applikation einen Table of Contents (ToC), eine Übersichtskarte, einen Nordpfeil, eine Maßstabsleiste und einen Copyright Text.

Am Ende des Wizards erfolgt die Anzeige aller festgelegter Konfigurationen und mit dem „Save Web App“ Button kann eine Web-GIS Anwendung erstellt werden. Zu diesem Zeitpunkt ist diese aber noch nicht auf dem Webserver deployed. Das .war File wird erstellt, wenn die Anwendung mit dem „Deploy“ Button im AGS veröffentlicht oder exportiert wird. Beim Exportieren ist es möglich, die .war Datei lokal abzuspeichern.

4.2 INTEGRATION IN DIE ENTWICKLUNGSUMGEBUNG

Die im ArcGIS Server Manager generierte .war Datei wird für die Weiterentwicklung der Webanwendungen in die Eclipse IDE importiert. Der Integrationsprozess ist in dieser Arbeit mit einem Workflow dargestellt. Zunächst aber werden die Architektur und die Komponenten der Testumgebung kurz vorgestellt.

4.2.1 ARCHITEKTUR DER TESTUMGEBUNG

Die Testumgebung besteht aus einer *Server Komponente* und einem *Entwicklungsrechner*. Auf der Server Komponente ist *der ArcGIS Server 9.3 Java Edition for Enterprise* installiert ist. Auf diesem befinden sich zudem ein Tomcat Webserver (Version 6.0) zum Deployen von LaFIS Web und eine Oracle Datenbank (11g) mit ArcSDE (9.3) Schnittstelle, in der die Geodaten für die Webapplikation vorgehalten werden.

Der verwendete Serverrechner hat folgende Hardwaremerkmale:
Prozessor: 2 x Intel® Xeon® E5310, je 4 x 1,6GHz, 4 GB RAM

Der *Entwicklungsrechner*, auf dem die Eclipse IDE mit seinen Plug-ins installiert ist, befindet sich im selben LAN wie die Serverkomponente. Die auf dem Server laufende LaFIS-Web Anwendung wird außerdem von diesem Rechner aus mit einem Browser aufgerufen und getestet.

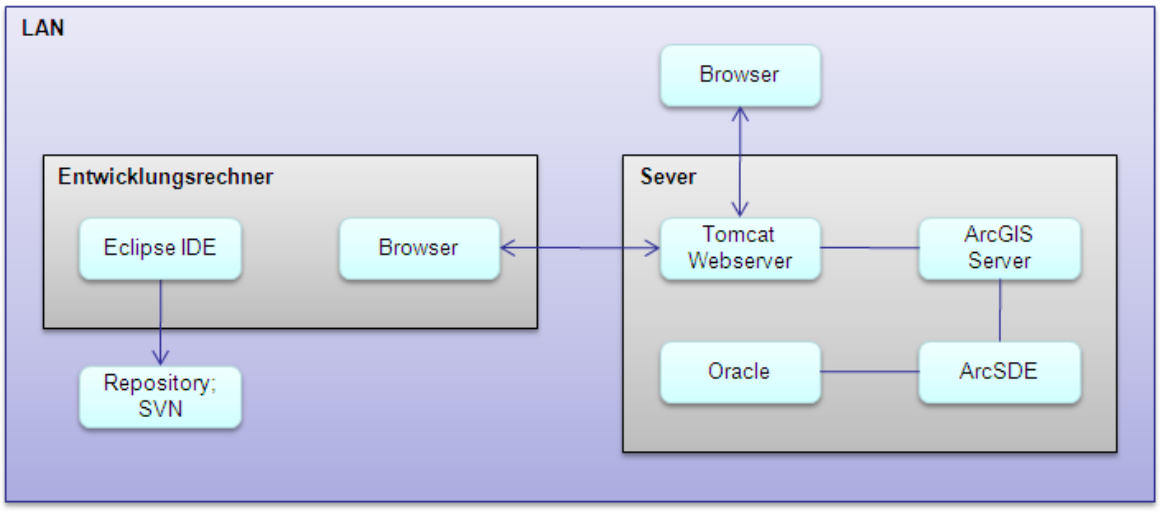


Abb. 18: Architektur der Testumgebung

Die Architektur der Testumgebung unterscheidet sich natürlich von der eines Produktivsystems. Zum Einen muss der Client mit einem Browser über das Internet auf die LaFIS-Web Applikation zugreifen können. Zum Anderen sind die Komponenten auf dem Server auf einzelne Rechner aufzuteilen, um bei mehreren Nutzern die Menge der eingehenden Anfragen bearbeiten zu können. Bei einer hohen Nutzerzahl ist sogar zu prüfen, ob nicht mehrere AGS SOC Rechner eingesetzt werden müssen.

4.2.2 Komponenten der Testumgebung

Als Entwicklungsumgebung wurde *Eclipse* in der Version 3.4.2 mit dem Projektnamen Ganymede verwendet. Es kam die Version für Java EE Entwickler zum Einsatz, welche die WebTools zur Entwicklung von Webapplikationen beinhaltet. Dabei sind die Java Bibliotheken für die JSF, JSP und Servlet Technologien bereits enthalten. Zudem beinhaltete die verwendete Eclipse IDE das Maven Plug-in und eine Tomcat Server Runtime Environment.

Im Testsystem wurde der *Apache Tomcat* Server in der Version 6.0.14 auf dem Entwicklungsrechner und auf dem Server, der AGS enthält, installiert und zum Deployen der LaFIS-Web-Applikation verwendet. Damit der Webserver in Verbindung mit Eclipse verwendet werden kann (z.B. zum Starten des Servers aus Eclipse oder zum Verwenden des Debug-Modus), ist er in der IDE als Server mit der entsprechenden Server Runtime Environment anzulegen. Dabei ist der Pfad zum Tomcat Installationsverzeichnis anzugeben.

Das *Build-Management Tool Maven* in der Version 2.1.0. ist in Eclipse IDE als Plug-in integriert. So kann jedes Eclipse Projekt für das Dependency Management mit Maven aktiviert werden. Zuvor müssen jedoch ein gültiges pom.xml File erstellt und alle relevanten Bibliotheken in einem Repository abgelegt sein, das in Maven registriert ist.

Sämtliche ESRI Bibliotheken, die in der ArcGIS Server Auslieferung enthalten sind (siehe Tab. 6), wurden zu dem Repository hinzugefügt. So liegen diese zentral für alle Entwickler vor und können ohne großen Aufwand zu jeder Zeit aktualisiert bzw. neue Versionen hinzugefügt werden.

4.2.3 INTEGRATIONSWORKFLOW

Der Integrationsworkflow für die Importierung der .war Datei in die Eclipse IDE gliedert sich grundsätzlich in vier Schritte:

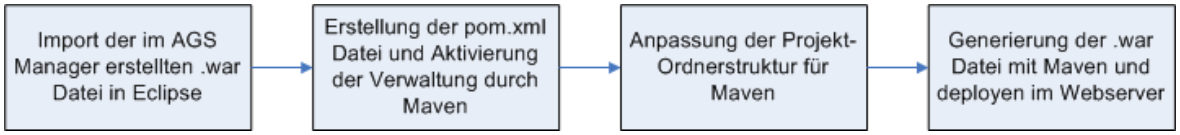


Abb. 19: Integrationsablauf in die Entwicklungsumgebung

1. Import der .war Datei in Eclipse
   Im ArcGIS Server Manager besteht die Möglichkeit, die LaFIS Web-Applikation als gepackte .war Datei zu exportieren. Die Importierung in die Eclipse IDE ist einfach mit der Import Funktion einer WAR Datei zu bewerkstelligen. Während des Import-vorgangs ist einfach der Namen des neuen Dynamic Web Projektes anzugeben.

2. Erstellung der pom.xml Datei und Aktivierung von Maven
   Das pom.xml ist eine einfache XML Konfigurationsdatei, in UTF-8 codiert und enthält folgende wichtige Bestandteile:
   - Applikationsname, Versionsnummer und Typ der Komprimierung (war)
   - Pfadkonfiguration zum Repository
   - Maven-Compiler-Plugin für die Kompilierung des Java Codes
   - Maven-WAR-Plugin zur Einbindung der Ressourcen für die Webanwendung
   - Dependencies zu den benötigten Bibliotheken

Die Abhängigkeiten (Dependencies) werden durch ein *groupId*, *artifactId* und *version* Attribut referenziert. Die Web Controls des ArcGIS Servers sind z.B. folgendermaßen eingebunden worden:

```
<dependencies>
      <dependency>
            <groupId>esri.arcgis</groupId>
            <artifactId>arcgis_webcontrols_ags</artifactId>
            <version>9.3</version>
      </dependency>
...
```

Listing 5: Dependeny Eintrag im pom.xml

Ist die pom.xml Datei mit allen notwendigen Elementen erstellt, kann das Eclipse Projekt für das Dependency Management mit Maven aktiviert werden. Bei diesem Vorgang lädt sich das Projekt die notwendigen Bibliotheken aus dem Repository und durchläuft einen Kompiliervorgang. Falls nicht alle notwendigen Bibliotheken als Dependencies deklariert sind, erkennt das Web Projekt die Klassen aus diesen Bibliotheken nicht und kompiliert deshalb nicht fehlerfrei. In diesem Fall sind die fehlenden Dependencies im pom.xml noch anzugegeben, bis das Projekt fehlerfrei kompiliert wird.

3. Anpassung der Projekt-Ordnerstruktur für Maven

   Neben der Bibliothekenverwaltung mit Dependencies ist Maven für die Kompilierung des Java Programmcodes und für das Packen der Web-Applikation zu einer .war Datei zuständig. Für das Packen einer Webanwendung erwartet Maven eine bestimmte Ordnerstruktur, die bereits in 3.4.1.2 dargestellt wurde. Die Ordnerstruktur wird im Eclipse Projekt Fenster durch Anlegen und Löschen von Ordnern und durch Verschieben der Inhalte geändert. Folgende Abbildung zeigt die finale Ordnerstruktur nach der Umstellung:

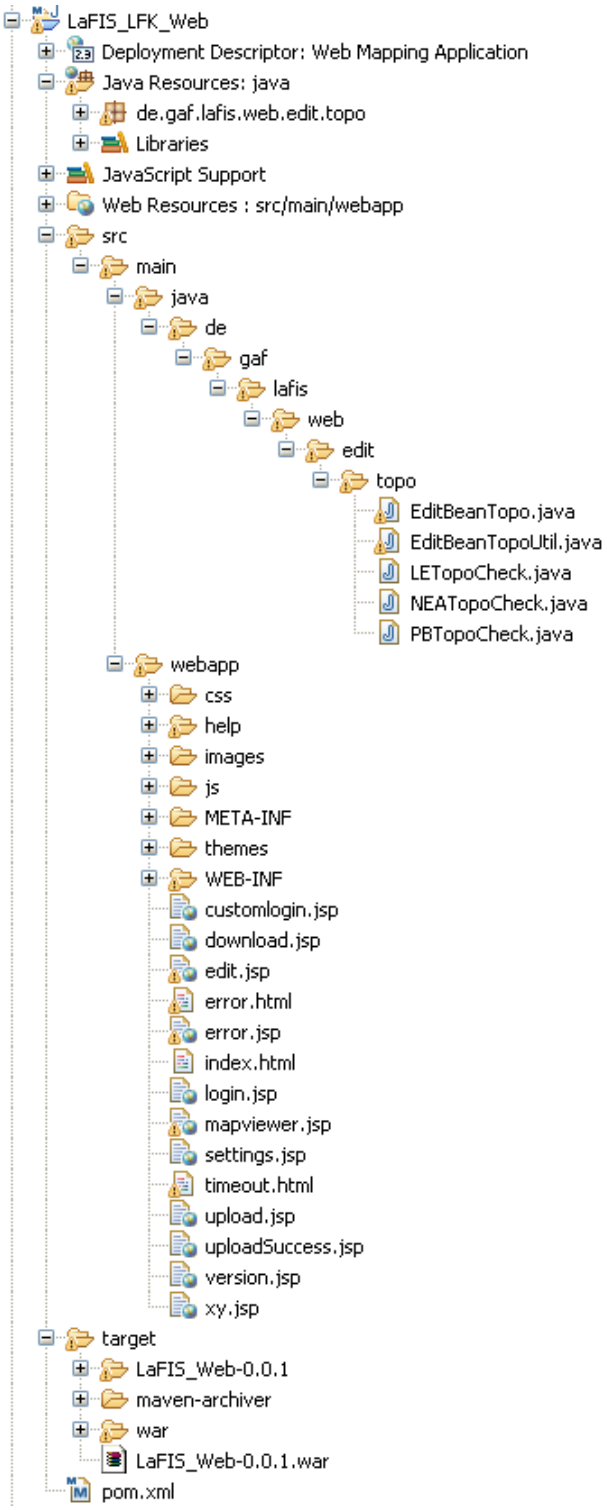


Abb. 20: Ordnerstruktur des LaFIS-Web Projektes in Eclipse

4. Generierung der .war Datei und Deployen im Webserver

   An dieser Stelle könnte nun mit der Weiterentwicklung begonnen werden. Zuvor empfiehlt sich jedoch ein Test des Packvorganges mit Maven und des Deployens auf dem Webserver, um die Funktionsfähigkeit der LaFIS-Web Anwendung zu testen.

Hierzu muss mit der Eingabeaufforderung von Windows zum LaFIS-Web Projektordner von Eclipse navigiert werden, indem sich auch das pom.xml File befindet. Der Befehl *„mvn install“* packt die LaFIS-Web Anwendung zu einer .war Datei. Läuft der Vorgang fehlerfrei ab, so wird diese in den Target Ordner im Eclipse Projekt kopiert.

Das Deployen der Webapplikation erfolgt mit dem Tomcat Manager, der über die URL *http://<servername>:8080/manager* aufgerufen werden kann. Je nachdem ob der Tomcat auf dem lokalen Rechner oder auf einem Server im LAN installiert ist, ist entweder „localhost“ oder der jeweilige Servername an der Stelle von <servername> anzugeben. Zum depolyen der Anwendung gibt es im Tomcat Manager einen speziell dafür vorgesehenen Dialog, bei dem nur die .war Datei auszuwählen ist, um diese anschließend per Knopfdruck im Webserver zu veröffentlich.

Nach dem erfolgreichen Deployen ist die Webanwendung für die URL *http://<servername>:8080/<.war-name>* aufrufbar.

4.3 Erweiterung des Editor Tasks

Der Editor Task wurde bereits im Kapitel 3.2.5 vorgestellt. Die *EditBean* Klasse ist die zentrale Klasse für den Editor Task, welche eine *BackingBean* ist. Sie regelt alle Editiervorgänge und enthält u.a. die Methode *„addPolygon“*, zum Editieren von Polygonen. Für die Erweiterung von Editierfunktionalitäten schlägt nun ESRI vor, diese Klasse zu erweitern, wobei einzelne Methoden auch überschrieben werden können. Auf diesem Konzept bauen auch die Topologieprüfungen in der LaFIS-Web Anwendung auf.

4.3.1 Regeln für den Topologie Check

Für den LaFIS-Web Prototyp ist es vorerst nicht vorgesehen, ein User Interface bereitzustellen, das dem Nutzer erlaubt, Einfluss auf das resultierende Polygon nach der Topologieprüfung zu nehmen. Das Treffen einer Auswahl, ob eine Überlappung vom neuen Polygon entfernt oder vom bestehenden abgeschnitten wird, ist für den Benutzer nicht möglich. Deshalb läuft die Prüfung der topologischen Beziehungen nach im Vorfeld festgelegten Regeln ab:

- Überlappungen von neu digitalisierten PB, LE oder NEA werden entfernt und verworfen. Das korrigierte Polygon ohne Überlappungen wird hinzugefügt.
- Liegt die Geometrie eines neuen PB, LE oder einer NEA innerhalb eines bestehenden PB (vollständige Überlappung), so wird diese neu digitalisiert, indem ihre Fläche von der Fläche des PB „ausgestanzt“ wird.
- Treten Multipart-Polygone auf, so wird die Geometrie des Rings mit der größten Fläche weiter verwendet. Die übrigen Multipart Flächen werden entfernt und verworfen.
- Trifft eine der in 3.1.2 aufgestellten topologischen Beziehungen nicht zu, so wird kein Polygon digitalisiert und der Editiervorgang abgebrochen. Der Editor Task befindet sich danach in dem Zustand, wie vor dem Digitalisieren des letzten Features.

4.3.2 Gesamtablauf des Editiervorgangs mit Topologie Check

Der Ablauf Topology Checks (siehe Abb. 21) knüpft an die Methode addPolygon() im Editor Task an. Dabei erfolgt die Übergabe des vom Benutzer gezeichneten Polygons. Auf Basis dieser Geometrie prüft der Topology Check die topologischen Beziehungen. Da jede Feature Klasse unterschiedliche Topologieprüfungen durchläuft, wird diese am Anfang der Methode bestimmt und anschließend deren spezifische Topologie geprüft. Verläuft die Prüfung der topologischen Beziehungen mit Erfolg, schleißt der Editor Task den Editiervorgang ab und fügt das neue Polygon dem WebContext hinzu. Ansonsten wird der Editiervorgang abgebrochen und das gezeichnete Polygon verworfen.

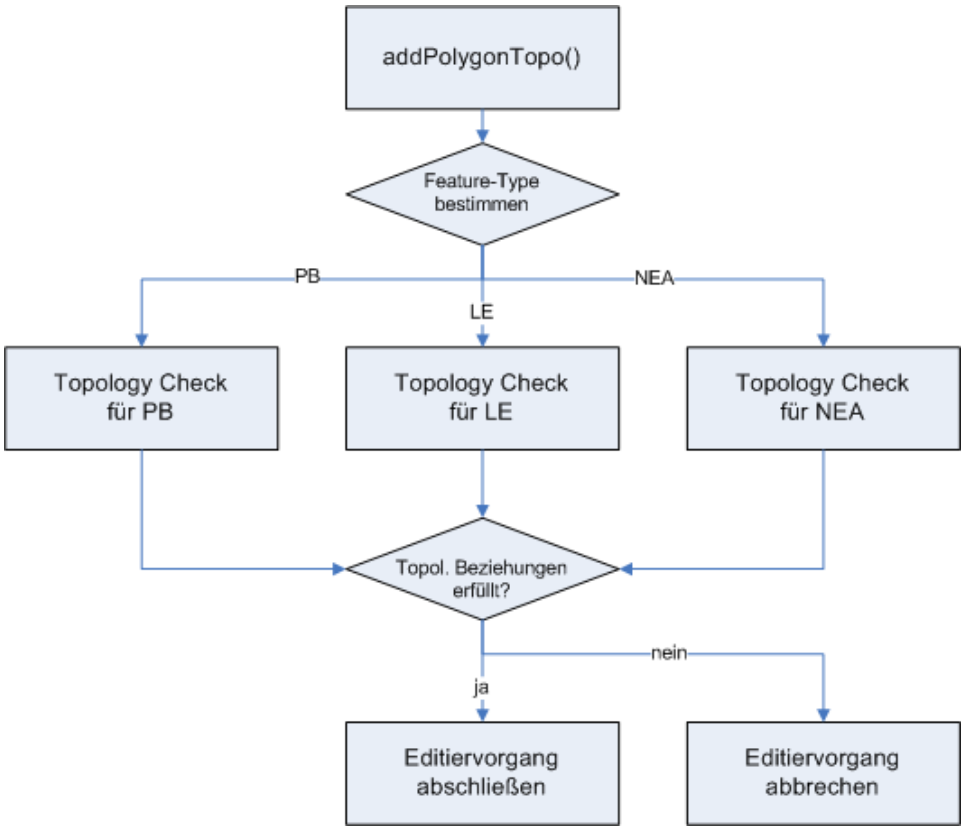


Abb. 21: Gesamtablauf des Topology Checks

Der Ablauf der topologischen Prüfungen für Feldblöcke (siehe Abb. 22) beginnt mit einem Test, ob die vom User gezeichnete PB Fläche innerhalb eines bestehenden Landschaftselements oder einer nicht förderfähigen Fläche liegt. Ist dies der Fall, führt dies zum Abbruch des Editiervorgangs (siehe topologische Beziehung (= t.B.) 2g).

Liegt der neu digitalisierte Feldblock innerhalb eines bestehenden Feldblocks (t.B. 2f), erfolgt weiterhin die Überprüfung der Mindestgröße. Ist diese erfüllt, wird die Fläche des neuen Feldblocks aus der bestehenden Fläche ausgestanzt und die neue Fläche dem WebContext hinzugefügt. Wenn nicht, führt auch dies zum Abbruch des Editiervorgangs.

Treffen beide oben beschriebenen Beziehungen nicht zu, wird die Geometrie des neuen Feldblocks auf Überlappungen mit Hilfe einer räumlichen Abfrage geprüft und gegebenenfalls mittels des topologischen Operator „difference“ entfernt (t.B. 2c). Durch diese Aktion können Multipart-Polygone entstehen, wenn sich die digitalisierte Geometrie über mehrere Flächen erstreckt. Wenn dies der Fall ist, erfolgt die Ermittlung der größten Ring-Fläche, die weiter verwendet wird, die anderen Multiparts werden entfernt (t.B. 1b). Abschließend erfolgt eine Überprüfung der Flächenmindestgröße. Ist diese groß genug,

fügt der Editor Task den neuen FB dem WebContext hinzu. Andernfalls erfolgt ein Abbruch des Editiervorganges.

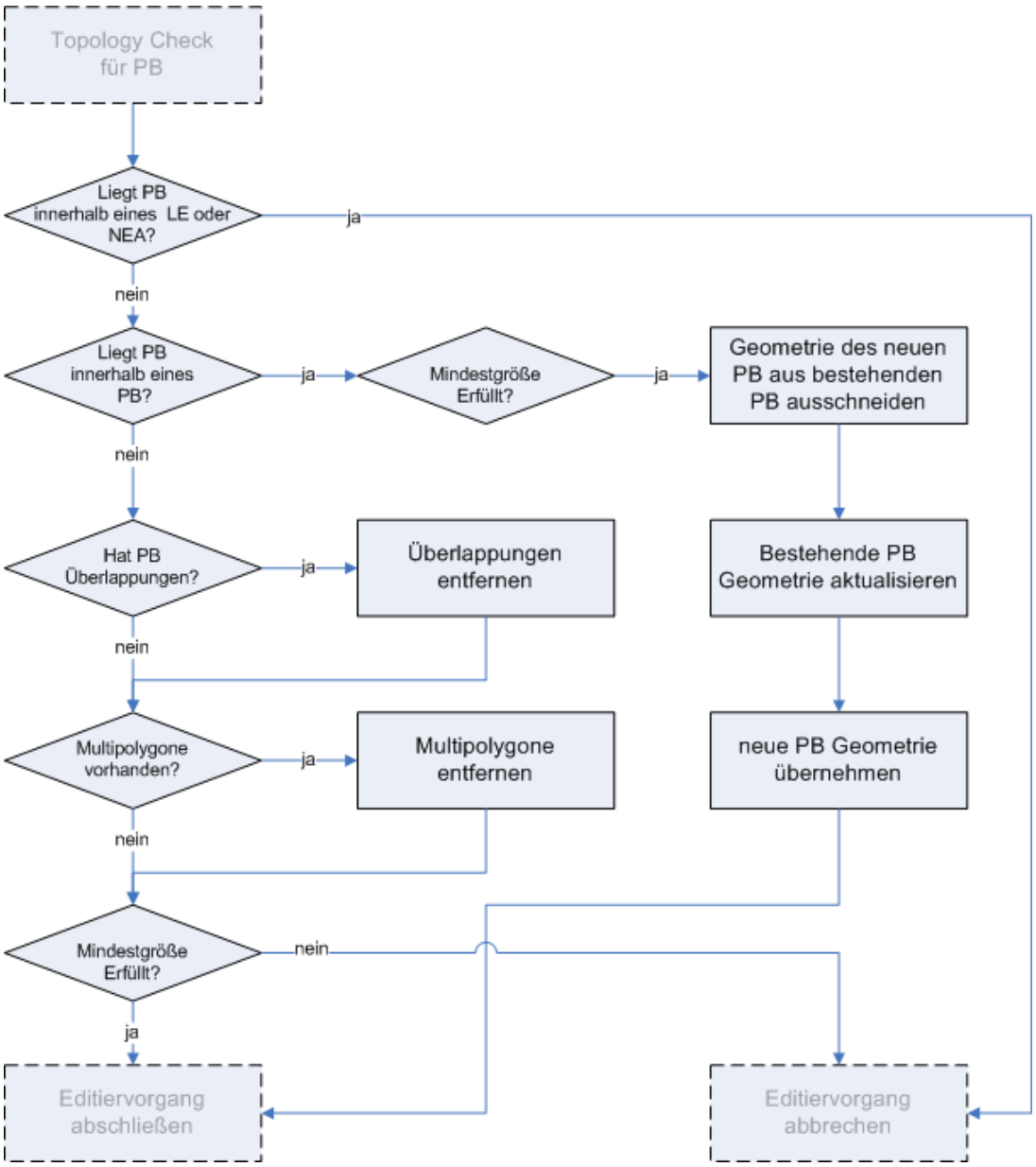


Abb. 22: Ablauf des Topology Checks für eine Feldblock Geometrie

Die Topology Checks für Landschaftselemente und nicht förderfähige Flächen laufen ähnlich ab und werden an dieser Stelle nicht näher betrachtet. Die Ablaufdiagramme zu jenen Topology Checks befinden sich im Anhang dieser Arbeit (siehe Anhang 1 & 2).

4.3.3 ÜBERSICHT DER VERWENDETEN ESRI KLASSEN

Für die Umsetzung der Topologie Funktionalitäten sind Klassen und Interfaces aus verschiedenen ESRI Bibliotheken notwendig. Diese sind in folgender Tabelle dargestellt:

ESRI Bibliothek	Paket	Klassen/Interfaces
ArcGIS Server Web ADF	com.esri.adf.web.ags.data	AGSLocalMapResource
	com.esri.adf.web.ags.data.edit.bean	EditBean
	com.esri.adf.web.data	WebContext
	com.esri.adf.web.data.geometry	WebGeometry
	com.esri.adf.web.faces.event	MapEvent
ArcGIS Web Services Java API	com.esri.arcgisws	Geometry, SpatialFilter, MapServerPort, RecordSet
ArcGIS ArcObjects Java API	com.esri.arcgis.geometry	Polygon, IGeometryBag, IEnumGeometry, IPolygon4, IGeometry5, IArea
	com.esri.arcgis.server	IServerContext

Tab. 6: Übersicht der ESRI Bibliotheken mit den verwendeten Klassen

4.3.4 IMPLEMENTIERUNG (DER TOPOLOGISCHEN FUNKTIONALITÄTEN)

Zur Implementierung der Prüfungen der topologischen Beziehungen erfolgte die Erstellung der neuen Klasse *EditBeanTopo*, welche die Klasse *EditBean* erweitert. Die `addPolygon()` Methode aus dieser Klasse könnte in der Klasse *EditBeanTopo* überschrieben werden. Damit aber weiterhin die Möglichkeit besteht, ein Polygon ohne Topologieprüfungen editieren zu können, ist dies nicht umgesetzt worden. Stattdessen erfolgte die Implementierung der neuen Methode `addPolygonTopo()`. Beiden Methoden wird beim Aufruf ein *MapEvent* übergeben, das von der Client Aktion (JavaScript) *„Polygon zeichnen“* ausgelöst wurde. Dieses *MapEvent* beinhaltet die Geometrie des am Bildschirm gezeichneten Polygons.

Nach dem Aufruf der `addPolygonTopo()` Methode wird in der *EditBeanTopo* mittels der Methode `getLayerID()` festgestellt, in welchem Layer ein Objekt erstellt wurde

(siehe Listing 6). Je nach Feature Klasse erfolgt die Instanziierung der entsprechenden Klasse *TopoCheck*. Durch Aufruf der `getCheckedFeature()` Methode erfolgt die Prüfung der topologischen Beziehungen in festgelegter Reihenfolge. Dabei werden Methoden aus der Klasse *EditBeanTopoUtil* verwendet, die allgemeingültige Methoden für alle TopoCheck Klassen bereithält. Die Prüfung der topologischen Beziehungen unterscheidet sich nur im Ablauf der Methoden und der eingesetzten Parameter. Auf den Ablauf und die Funktionsweise der implementierten Methoden wird in den nächsten Punkten näher eingegangen.

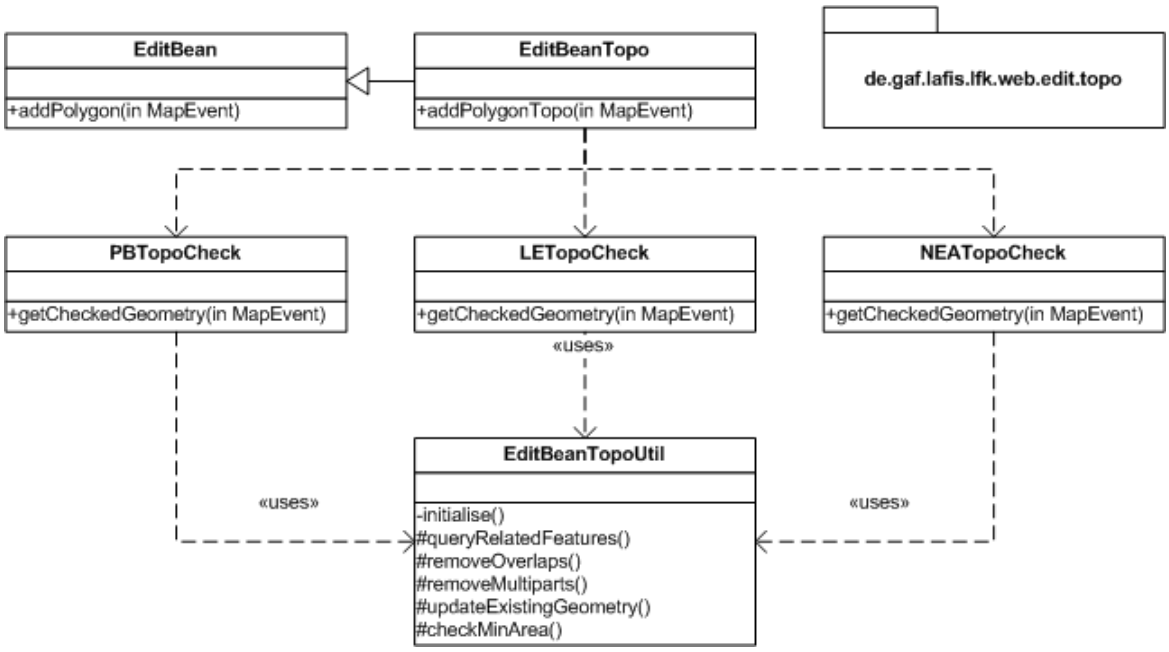


Abb. 23: Klassenübersicht des Topology Checks

```
int workingLayerID = getLayerID();

if (workingLayerID == 1) {
      leTopoCheck = new LETopoCheck(event, webGeometry editLayersMap,
            workingLayerID);
      checkedFeature = leTopoCheck.getCheckedGeometry();
} else if (workingLayerID == 2) {
      neaTopoCheck = new NEATopoCheck(event, webGeometry, editLayersMap,
            workingLayerID);
      checkedFeature = neaTopoCheck.getCheckedGeometry();
} else if (workingLayerID == 3) {
      pbTopoCheck = new PBTopoCheck(event, webGeometry, editLayersMap,
            workingLayerID);
      checkedFeature = pbTopoCheck.getCheckedGeometry();
}
```

Listing 6: Bestimmung des TopoChecks durch die Layer ID in der EditBeanTopo Klasse

Der weitere Verlauf der Arbeit beschreibt die Implementierung der Klasse *PBTopoCheck*, also die Prüfung der topologischen Beziehungen für Feldblöcke (siehe Abb. 22). Die Implementierung der beiden anderen TopoCheck Klassen ist ähnlich und wird daher nicht näher erläutert.

4.3.4.1 Liegt PB innerhalb eines LE oder NEA?

Diese Fragestellung klärt die `queryRelatedFeatures()` Methode. Hierzu ist der Methode lediglich die gezeichnete Geometrie als *Filter-Geometrie* (= newAGSFeature) und der *räumliche Relationsparameter* `esriSpatialRelWithin` (= spRelation) zu übergeben. Diese beiden Parameter sind Elemente eines räumlicher Filters (SpatialFilter siehe Listing 7), der anschließend in einer räumlichen Abfrage `queryFeatureData()` (siehe Listing 8) auf die Ebenen der LE und NEA angewendet wird, um die in Beziehung stehenden LE und NEA Objekte zu ermitteln. Das Ergebnis dieser Methode ist eine Liste mit den Geometrien, die auf die Bedingung des räumlichen Filters zutreffen. Ist die Liste leer, so liegt der gezeichnete Feldblock nicht innerhalb eines Landschaftselements oder einer nichtförderfähigen Fläche. In diesem Fall erfolgt die Überprüfung auf Überlappungen, ansonsten ein Abbruch des Editiervorgangs.

```
SpatialFilter filter = new SpatialFilter();
filter.setFilterGeometry(newAGSFeature);
filter.setSpatialRel(spRelation);
filter.setSearchOrder(EsriSearchOrder.esriSearchOrderSpatial);
filter.setWhereClause("");
filter.setGeometryFieldName("");
filter.setSpatialReferenceFieldName("");
filter.setSpatialRelDescription("");
```

Listing 7: Räumlicher Filter zur Ermittlung der in Beziehung stehenden Features

Die `queryRelatedFeatures()` Methode verwendet zur räumlichen Abfrage den *MapServerPort* aus der ArcGIS Web Services API (siehe 3.3.2). Dadurch arbeitet die Web-Applikation nur mit den „grobkörnigen“ ArcObjects über einen Webservice. Die Attribute des Filters werden im Webserver gesetzt und nur bei der räumlichen Abfrage `queryFeatureData()`erfolgt ein Aufruf zum ArcGIS Server, was den Datenverkehr im Netzwerk zwischen Web- und GIS Server verringert (siehe 3.3.2).

```
RecordSet recordSet = mapServer.queryFeatureData(mapName, layerID,
filter);
```

Listing 8: Abfrage der in Beziehung stehenden Features über den Webservice MapServerPort

4.3.4.2 Liegt der PB innerhalb eines anderen PB?

Die Umsetzung dieser Fragestellung ist identisch mit der zuerst genannten, jedoch mit dem Unterschied, dass die räumliche Abfrage sich nur auf den Feldblock Layer bezieht. Ist in der Ergebnisliste eine Geometrie vorhanden, so befindet sich der gezeichnete Feldblock innerhalb eines anderen Feldblocks. Im Falle dieser Situation sind folgenden Schritte durchzuführen:

- Mindestgröße erfüllt? (wird später erläutert)
- Geometrie des neuen PB aus bestehenden PB ausschneiden
- Bestehende PB Geometrie aktualisieren
- Neue PB Geometrie übernehmen

Die letzten drei Schritte sind im LaFIS-Web Prototyp noch nicht programmiertechnisch umgesetzt worden.

4.3.4.3 Hat PB Überlappungen?

Auch bei dieser Fragestellung wird die Methode `queryRelatedFeatures()` aus der Klasse *EditBeanTopoUtil* eingesetzt, allerdings mit dem räumlichen Relationsparameter `esriSpatialRelIntersects`. Die räumliche Abfrage erfolgt über alle drei editierbaren Layer. Enthält die Ergebnisliste Geometrien, so sind Überlappungen vorhanden. Diese müssen in einem weiteren Schritt entfernt werden.

4.3.4.4 Überlappungen entfernen

Das Entfernen von Überlappungen erfolgt durch den Aufruf der `removeOverlaps()` Methode aus der Klasse *EditBeanTopoUtil*. Das gezeichnete Polygon und die Liste der überlappenden Features werden dieser Methode übergeben. Für das Arbeiten mit topologischen Operatoren sind jedoch Objekte der ArcObjects *Geometry* Klassen, also „feinkörnige“ ArcObjects notwendig. Deshalb müssen das gezeichnete Polygon und

diejenige aus der Abfrageliste in *ArcObjects Polygone* umgewandelt werden. Da die ArcObjects *Geometry* Klassen die Interfaces von *ITopologicalOperator* implementieren, stehen für Polygon Objekte Methoden mit topologischen Operationen zur Verfügung.

So z.B. verwendet das gezeichnete Polygon zur Entfernung der Überlappungen die `difference()` Methode (siehe Listing 9) aus der Polygon Klasse. Der Methode wird ein Polygon übergeben, mit dem die Verschneidung stattfinden soll. Dieses Polygon stammt aus der Liste der überlappenden Features. Jedes Polygon aus der Liste wird der Reihe nach mit dem neuen Polygon verschnitten, bis am Ende ein überlappungsfreies Polygon vorhanden ist. Dieses gibt die `removeOverlaps()` Methode am Ende zurück.

```
Polygon newFeature = (Polygon) feature.difference(geomToCompare);
```

Listing 9: Verschneidung zweier Polygone mit der difference() Methode

4.3.4.5 Multipolygone vorhanden?

Die *Polygon* Klasse ist eine *„Collection of Rings“*, d.h. sie kann aus mehreren Ring-Geometrien bestehen. Besteht ein Polygon aus mehr als einem Ring, ist es ein Multipart-Polygon. Durch das Entfernen von Überlappungen kann es vorkommen, dass das zurückgegebene Polygon ein Multipart Polygon ist. Die `getGeometryCount()` Methode aus der *EditBeanTopoUtil* Klasse überprüft, ob es sich bei dem untersuchten Feature um ein Multipart Objekt handelt. Dies trifft zu, wenn der ermittelte Wert größer als 1 ist.

4.3.4.6 Multipolygone entfernen

Multipart Polygone sind als Feldblock Geometrien nicht zugelassen (siehe Integritätsregel (IR) 1b). Im Punkt 4.3.1 ist festgelegt worden, dass die Geometrie mit der größten Fläche für das neue Polygon übernommen wird und die anderen Flächen verworfen werden. Die Methode `removeMultiparts()`aus der EditBeanTopoUtil Klasse ermittelt die einzelnen Flächen der Ringe mit der Methode `getConnectedComponentBag()`und verglichet sie miteinander. Die Flächengröße erhält man mit der Methode `getArea()` aus der *Polygon* Klasse. Die Methode `removeMultiparts()` gibt am Ende die Geometrie mit der größten Fläche als Polygon zurück.

4.3.4.7 Mindestgröße erfüllt?

Am Ende des Topologie Checks überprüft die Methode `checkMinArea()`, ob die Mindestgröße für eine Feldblock Geometrie erfüllt ist. Dies geschieht auch mit der `getArea()` Methode aus der *Polygon* Klasse. Ist die Mindestgröße nicht erfüllt, gibt die `checkMinArea()` Methode `null` zurück und bricht somit den Editiervorgang ab.

```
Polygon checkMinArea(Polygon geomToCheck, int workingLayerID) {
      Polygon checkedGeom = null;
      if (workingLayerID == 3 && geomToCheck.getArea() < 10) {
            checkedGeom = null;
      } else if ((workingLayerID == 1 || workingLayerID == 2) &&
      geomToCheck.getArea() < 1) {
            checkedGeom = null;
      } else {
            checkedGeom = geomToCheck;
      }
}
```

Listing 10: Prüfung der Mindestgröße mit der checkMinArea() Methode

5 ERZIELTE ERGEBNISSE

Mit den vorgestellten Technologien, Werkzeugen und Methoden wurde ein Web-GIS Client mit Editierfunktionalitäten erstellt. Die Editierwerkzeuge bieten eine Vielzahl von Tools für das Erstellen und Bearbeiten von geographischen Objekten. Zum Editieren unter Berücksichtigung von topologischen Beziehungen ist aber noch ein weiteres Tool dem bestehenden Editor Task hinzugefügt worden.

5.1 DER WEB-GIS CLIENT

Mit dem ArcGIS Manager war es möglich, die Web-GIS Applikation LaFIS-Web mit seinen Editierwerkzeugen ohne Programmierarbeiten zu erstellen. Der browserbasierte Client (siehe Abb. 24) setzt sich aus folgenden Elementen zusammen:

- **Kartenfenster**
  Enthält das Kartenfenster in dem der User interagieren kann, sowie die *Maßstabsleiste*, *Nordpfeil* und den *Copyrightvermerk*
- **Legende / Table of Contents (ToC)**
  Zeigt die im Kartenfenster dargestellten *Kartenebenen*, die sichtbar oder unsichtbar geschaltet werden können. Das Context-Menü eines Layereintrages ermöglicht das Zoomen zur räumlichen Ausdehnung der Kartenebene.
- **Result Panel**
  Zeigt die Ergebnisse des *Search Attributes* Tasks und des *Identify* Tools an.
- **Werkzeugleiste**
  Enthält Werkzeuge zum Navigieren im Kartenfenster und stellt Messwerkzeuge sowie einen Link für eine Übersichtskarte zur Verfügung.
- Link zum **User Interface für Attributsuche**
  Erlaubt die Suche nach Attributdaten, die zuvor im ArcGIS Manager festgelegt wurden.
- Link zum **User Interface für den Editor Task**
  Nutzeroberfläche zum Editieren von geographischen Features und deren Attribute.

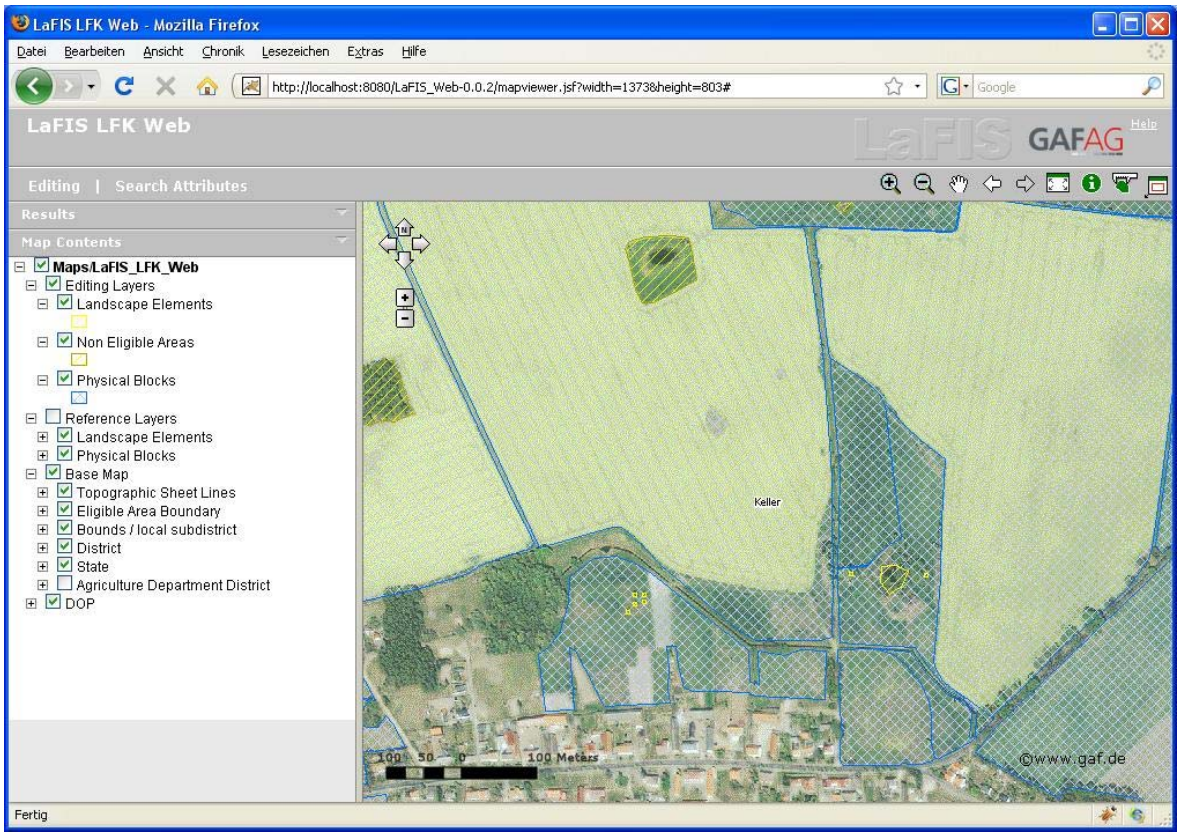


Abb. 24: Der LaFIS-Web Client

Die LaFIS-Web Anwendung und die oben genannten Kartenelemente und Tasks sind in der ArcGIS Manager Oberfläche konfiguriert worden. Ebenfalls der Editor Task, der das User Interface und die Werkzeuge für das Editieren von geographischen Features bereithält.

Das ***Editor Task User Interface*** (siehe Abb. 25) enthält die vier Abschnitte *„Editing“*, *„Create Feature“*, *„Edit Feature“* und *„Attributes“*.

Im Bereich *Editing* erfolgt zunächst die Auswahl des zu editierenden Layers. Da die LaFIS-Web Anwendung über eine lokale GIS-Ressource auf die Datenquelle zugreift, enthält die Werkzeugleiste einen *„save edits“* und *„discard edits“* Button. Bei einer Internet Verbindung zur GIS-Ressource erfolgt die Speicherung der Editierungen sofort in der Datenbank. Aber durch die lokale Verbindung werden die neu digitalisierten Features zunächst dem WebContext hinzugefügt und erst durch den *„save edits“* Button in die Datenbank gespeichert. Die Tools *„undo“* und *„redo“* werden durch eine versionierte Datenbank ermöglicht. Weitere Werkzeuge im Abschnitt *Editing* sind *„select feature“*, *„deselect“*, *„show vertices“* und *„enter X/Y coordinates“*.

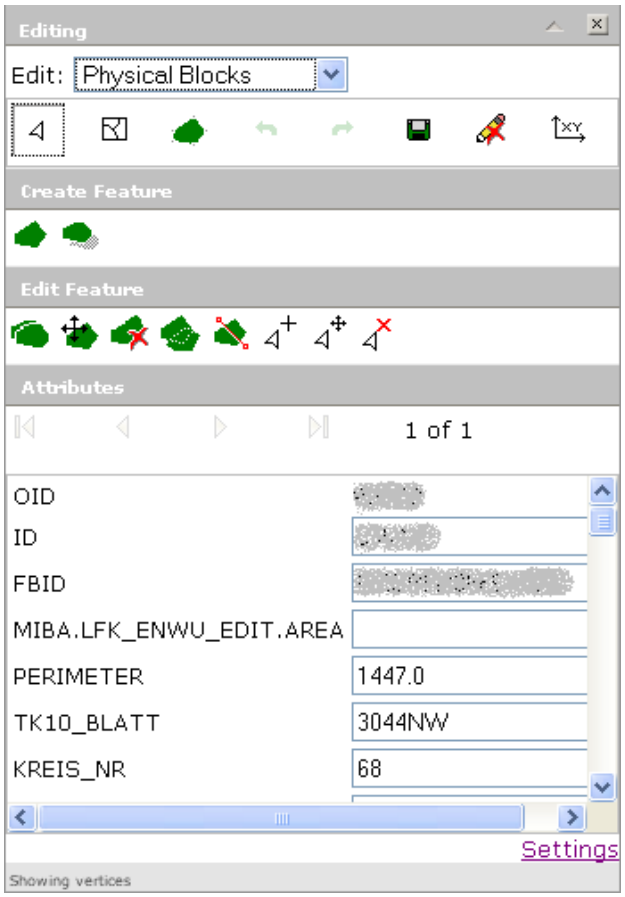


Abb. 25: Das Editor Task User Interface

Die Leiste „*Create Feature*“ enthält neben dem standardmäßigen Tool „*add polygon*“ das neue entwickelte Tool „*add Polygon with topology check*“. Durch Auswahl dieses Buttons kann der User im Kartenfenster ein neues Polygon zeichnen. Die Methoden aus der Klasse *EditBeanTopoUtil* prüfen in der entsprechenden Klasse *TopoCheck* das gezeichnete Polygon. Dabei werden auftretende topologische Fehler bereinigt und anschließend das korrigierte Polygon dem WebContext hinzugefügt.

Die „*Edit Feature*“ Werkzeugleiste bietet nach Auswahl eines Features dem User die Möglichkeit, dieses zu *kopieren*, *verschieben*, zu *löschen*, zu *vereinen*, zu *teilen* und *Stützpunkte hinzuzufügen*, *zu verschieben* oder zu *löschen*.

Das „*Attribute*“ *Feld* bietet die Möglichkeit, die Attribute eines ausgewählten Features zu bearbeiten. Die Änderungen werden während der Eingabe validiert und dem WebContext hinzugefügt. Erst nachdem der Command Button „*save edits*“ ausgeführt wurde, gelangen die Änderungen in die Datenbank.

Der Link *„Settings“* im unteren rechten Teil des Editor Task UI verweist auf ein neues Fenster, das Einstellungen zur Snapping-Funktion und Farbeinstellungen für selektierte Features bereitstellt.

Die im AGS-Manager erstellte LaFIS Applikation verwendete zu Beginn eine von ESRI *vordefinierte Designvorlage (sog. „Theme“)*, das jedoch an das GAF AG und LaFIS Erscheinungsbild angepasst werden musste. Das neue *Theme* für die GAF AG baut auf den vordefinierten Themes von ESRI auf, verwendet aber andere Hintergrundfarben und –Bilder, was durch Änderung der CSS Dateien realisierbar war. Ebenfalls wurden sämtliche Bilder bzw. Icons für die Werkzeuge ausgetauscht. Diese Änderungen erzielten ein Look & Feel, das der GAF AG und der bisherigen LaFIS Anwendung entsprach.

5.2 DAS EDITING TOOL MIT TOPO CHECK

Das Kapitel 3.5 stellte für die Integration der topologischen Prüfungen Beziehungen Konzepte vor, wie diese in eine ArcGIS Server Web-GIS Anwendung integriert werden können. Die darauffolgende Umsetzung (Kapitel 4) zeigte die notwendigen Arbeitsschritte und Werkzeuge, sowie die Implementierung durch die Erstellung des neuen Tools „*add polygon with topology check*“.

Vor der Umsetzung sind im Kapitel 3.1.2 die Integritätsregeln und topologischen Beziehungen der LaFIS Test Daten dargestellt. Welche dieser Forderungen im neu entwickelten Tool des Prototypen umsetzbar waren, zeigt folgende Tabelle im Überblick:

Nr.	Regel/Beziehung	Umsetzung
1a	Es sind nur Polygone für Referenz-flächen zulässig.	Diese Bedingung wird durch Polygon Feature Klasse sichergestellt.
1b	Es sind keine Multi-Part Polygone erlaubt	Umgesetzt durch die Methode `removeMultipart()`
1c	Polygone haben eine Minimalfläche	Umgesetzt durch die Methode `checkMinAra()`
1d	Die Genauigkeitsangabe der Koordinaten liegt bei 1mm	Der Wert wurde durch Angabe der „*XY tolerance*“ in der Polygon Feature Klasse festgelegt.
2a	Polygone dürfen sich nicht selbst überschneiden	Der Editor Task lässt keine Self-Intersections zu.

2b	Es sind keine übereinanderliegende Stützpunkte erlaubt	Nicht umgesetzt, könnte bei der Definition einer Topologie in der DB festgelegt werden.
2c	Referenzobjekte dürfen sich nicht überlappen	Umgesetzt durch die Methoden `queryRelatedFeatures()` und `removeOverlaps()`
2d	Es sind nur Lücken zwischen den Referenzobjekten erlaubt, die größer als 10cm sind	Nicht umgesetzt
2e	Aneinander liegende Referenzobjekte teilen sich gemeinsam die Grenzlinie	Durch die Verschneidung mit dem Topologischen Operator „*difference*“ entsteht eine gemeinsame Grenzlinie.
2f	Ein PB kann innerhalb eines anderen PB liegen	Das Editieren eines neuen PB innerhalb eine bestehenden PB ist noch nicht möglich, da die Ausschneidefunktion noch nicht implementiert ist.
2g	Ein PB kann NICHT innerhalb eines LE oder NEA liegen	Umgesetzt durch die Methode `queryRelatedFeatures()` mit dem Relate Operator „*within*“
2h	Ein LE kann komplett innerhalb eines PB liegen, muss aber mindestens einen gemeinsamen Stützpunkt mit einem PB haben	Umgesetzt durch die Methode `queryRelatedFeatures()` mit dem Relate Operator „*touches*“.
2i	LE können eine gemeinsame Grenzlinie mit eine anderen LE oder einer NEA haben	Durch die Verschneidung mit dem Topologischen Operator „*difference*“ entsteht eine gemeinsame Grenzlinie.
2j	NEAs können innerhalb oder außerhalb von PB, LE oder anderen NEA liegen	Erfordert keine Umsetzung

Tab. 7: Übersicht der festgelegten und umgesetzten Integritätsregeln und topologischen Beziehungen

Die folgenden Darstellungen zeigen Editiersituationen, die topologische Fehler hervorrufen würden. Daneben ist das vom Topology Check korrigierte Polygon dargestellt.

Situation 1: Überlappungssituation

Das vom Benutzer gezeichnete Polygon überlappt einen vorhandenen Feldblock. Die Überlappung wird mit der Prüfung „Hat PB Überlappungen?“ festgestellt und mit der Aktion „Überlappungen entfernen“ entfernt.

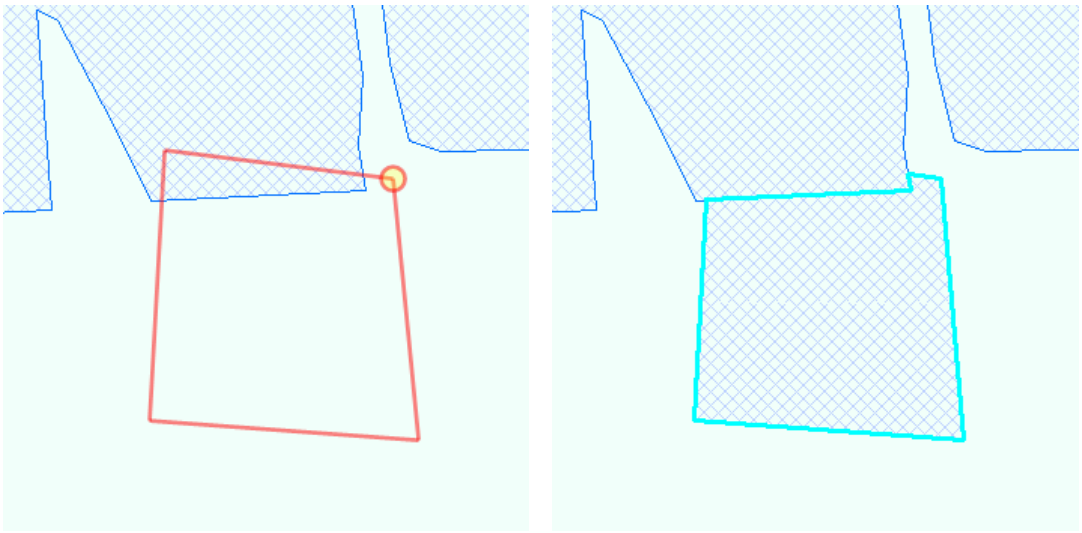


Abb. 26: Darstellung einer Überlappungssituation

Situation 2: Multipart Polygone

Der Benutzer zeichnet ein Polygon, das mehrere vorhandene Feldblöcke überlappt. Durch die Entfernung der Überlappungen entstünde ein Multipart Polygon. In dieser Situation wird die Prüfung „Mulitpolygone vorhanden?“ und die Aktion „Multipolygone entfernen“ angestoßen. Der Benutzer erhält den Polygonteil mit der größten Fläche als Ergebnis.

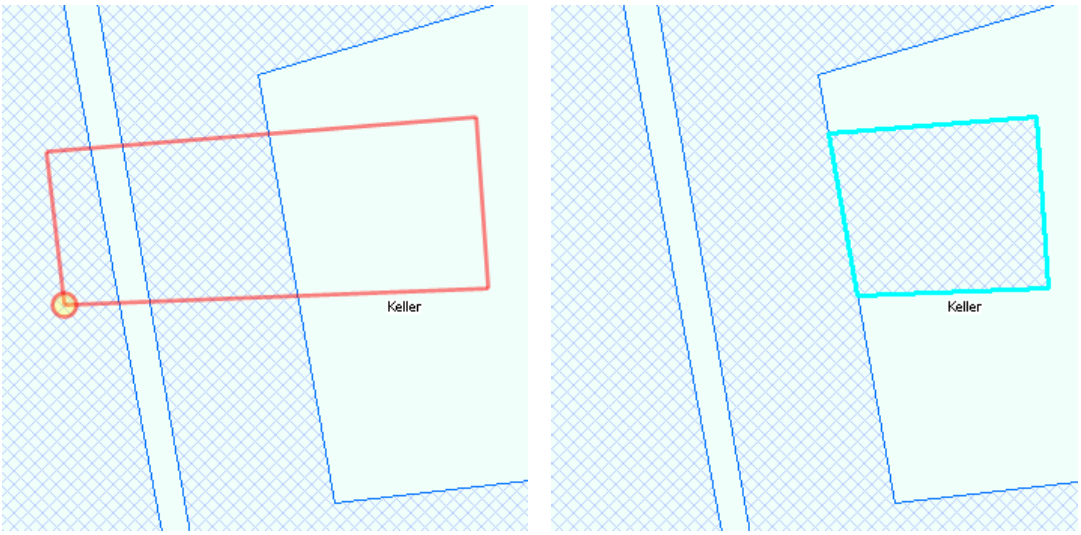


Abb. 27: Darstellung einer Multipart Polygon Situation

5.3 Analyse der Ergebnisse

Entscheidend für die LaFIS-Web Anwendung ist die *Benutzerfreundlichkeit* des Web-GIS Clients und dessen GIS-Werkzeuge, sowie das *Antwort-Zeit-Verhalten* des Web- und GIS-

Servers bei Benutzeranfragen. Die Benutzeroberfläche der Anwendung lehnt sich stark am Aufbau einer herkömmlichen Desktop Anwendung an und es stehen die standardmäßigen Tools für das Navigieren in der Karte zur Verfügung. Auch der *Table of Contents* bietet die wesentlichen Funktionalitäten. Dadurch, dass die Tasks im ArcGIS Server Web ADF AJAX implementieren, sind asynchrone Anfragen möglich. Ein Warten auf das Ende einer ausgeführten Aktion ist dadurch nicht nötig.

Das *Antwort-Zeit-Verhalten* beschreibt den Zeitraum ausgehend von einer Client-Aktion bis hin zur Darstellung des Anfrageergebnisses am Bildschirm. Für eine benutzerfreundliche Anwendung dürfen keine zu langen Wartezeiten beim Navigieren in der Karte und während der Erfassung von Polygonen entstehen. Die Geschwindigkeit des Editiervorgangs in LaFIS-Web ist jedoch noch nicht zufriedenstellend, da dieser in der Testumgebung (siehe 4.2.1) durchschnittlich ca. *fünf bis sechs Sekunden* dauert. Die Messung der Dauer beginnt mit dem Doppelklick auf das Polygon und endend mit der Darstellung des neuen gezeichneten Polygons am Bildschirm.

Um besser einschätzen zu können, welche Funktionen im Editor Task die langen Wartezeiten verursachen, sind dessen Methodenaufrufe einem Performacetest in der Testumgebung (siehe 4.2.1) unterzogen worden. Mit einem Zeitstempel am Anfang und am Ende eines jeden Methodenaufrufs lässt sich dessen Dauer ermitteln. Die folgende Tabelle zeigt, in Abhängigkeit des verwendeten Editiertools und der gegebenen Editiersituation, den Zeitaufwand für bestimmte Abschnitte im Editor Task. Dabei ist zu berücksichtigen, dass sich die Zeitstempel im Quellcode der Webapplikation befinden, also im Webserver. Das hat zur Folge, dass sich die Zeitangaben auf die Kommunikation zwischen Web- und GIS Server beziehen. Die benötigte Zeit zum Client ist dabei nicht berücksichtigt.

Editor Task / Abschnitt / Editier-Situation	Gesamtdauer Editiervorgang */**	Dauer des Topologie Checks **	Dauer einzelner Funktionen **
Feldblock ohne Überlappungen (mit „*add Polygon*“ Tool von ESRI)	2631 ms	-	-

Feldblock ohne Überlappungen (mit „*add Polygon with topology check*“ Tool)	3597 ms	922 ms	a = 349 ms b = 544 ms
Feldblock überlappt einen anderen Feldblock (mit „*add Polygon with topology check*“ Tool)	3623 ms	972 ms	a = 354 ms b = 501 ms c = 41 ms
Feldblock überlappt mehrere Feldblöcke, so dass ein Multipolygon entsteht. (mit „*add Polygon with topology check*“ Tool)	3790 ms	1141 ms	a = 369 ms b = 612 ms c = 131 ms d = 16 ms
* Ab Beginn der addPolygon() Methode im Editor Task bis nach dem aktualisieren des WebContext. ** Durchschnittswert aus 5 Messungen/Editiervorgängen a = Spatial Query „within“ b = Spatial Query „touches“ c = Methode removeOverlaps() d = Methode removeMultiparts()			

Tab. 8: Auswertung des Performancetests

In der Performance Tabelle ist zu sehen, dass der Editiervorgang “*add Polygon*” von ESRI im Webserver bereits knapp drei Sekunden dauert. Der Topologie Check verlängert das Editieren eines Polygons zusätzlich um ca. eine Sekunde. Dabei ist kein großer Unterschied zu erkennen, ob nur Überlappungen oder zusätzlich noch Multipart Polygone vorhanden sind. Auffallend ist nur, dass die räumlichen Abfragen, welche mit Web Services implementiert sind, relativ lange dauern. Ob diese Methoden performanter abliefen, wenn sie mit der ArcObjects API implementiert wären, müsste ein weiterer Performance Test zeigen.

Der jetzige Prototyp ist derzeit für eine Benutzerzahl von bis zu sechs gleichzeitigen Usern konfiguriert (siehe 4.1.2). Um eine Entscheidung treffen zu können, welche Hard- und Software Ressourcen für eine produktive Anwendung erforderlich sind, müssen weitere Performancetests durchgeführt werden. Das Verhalten der Serverauslastung, der LaFIS-Web Applikation und dessen Editierwerkzeuge sollte dabei insbesondere bei hohen Benutzerzahlen geprüft werden.

Eine Alternative für höhere Nutzerzahlen wäre eine andere Konfiguration des Map Services im ArcGIS Manager, indem als Verbindungstyp ein „pooled Service“ (siehe

3.2.3) festgelegt würde. Dabei entfällt aber die Möglichkeit, für den Datenbestand eine datenbankseitige Topologie zu definieren und die „*undo*“ und „*redo*“ Funktionen im Editor Task stünden auch nicht mehr zur Verfügung. Eine endgültige Entscheidung über die gewählte Konfiguration muss bei der Planung eines konkreten Systems in Abhängigkeit an die Systemanforderungen getroffen werden.

6 ZUSAMMENFASSUNG, DISKUSSION UND AUSBLICK

6.1 ZUSAMMENFASSUNG

Auf Basis der ArcGIS Server Technologie und mit dem AGS-Manager wurde eine voll funktionsfähige Web-GIS Anwendung mit umfangreichen Editierfunktionalitäten erstellt. Das Einbinden der Editierwerkzeuge in die Web-GIS Anwendung war mit vordefinierten Tasks möglich. Dies geschah ohne Programmieren und ohne Verwendung weiterer Tools, sozusagen eine Applikation „*out of the box*“ mit dem AGS-Manager.

Die daraus entstandene Anwendung war im Grunde einsatzfähig und mit umfangreichen GIS- und Editier Tools ausgestattet, jedoch mussten diese für die Prüfung von topologischen Beziehungen erweitert werden. Zudem erforderte das Look & Feel eine Anpassung, um dem Erscheinungsbild der GAF AG und der Software LaFIS zu gleichen. Dazu war die Integration der im AGS-Manager erstellten Applikation in eine Entwicklungsumgebung notwendig, was anhand eines Workflows in dieser Arbeit dargestellt wurde.

Für die Integration der Prüfung der topologischen Beziehungen sind im Kapitel 3.5 Konzepte vorgestellt worden, wie diese in eine ArcGIS Server Web-GIS Anwendung integriert werden können. Die darauffolgende Umsetzung zeigte die Variante der Erweiterung des Editor Tasks. Dafür sind die notwendigen Arbeitsschritte, die verwendeten Werkzeuge, die Struktur für die Gesamtlösung und ausgewählte Implementierungsbeispiele dargestellt worden. Abschließend erfolgte eine Darstellung des Web-GIS Clients und der implementierten Funktionalitäten zur Prüfung der topologischen Beziehungen.

6.2 DISKUSSION

Web-GIS Anwendungen, die als einfache oder spezialisierte georeferenzierte Auskunftssysteme dienen, sind weit verbreitet, sowohl als Anwendungen für die Öffentlichkeit, als auch in Form von Firmen- oder Behördenlösungen. Die Anwendungen mit umfangreichen Editierfunktionalitäten sind dagegen wesentlich seltener und dabei ist meist nur das Editieren von einfachen Geometrien, den so genannten Simple Features

möglich. Bei der Erfassung von topologisch richtigen Geo-Objekten stößt man jedoch sehr schnell an die Grenzen der zur Verfügung stehenden Möglichkeiten. Eine Umsetzung ist nur mit größerem Programmieraufwand zu erreichen, wobei sich schnell die Frage stellt, ob browserbasierte Web-GIS Clients für aufwändige GIS Fragestellungen geeignet sind. Interessant sind solche Anwendungen aber durchaus, da sie eine kostengünstige Alternative zu proprietären Desktop Anwendungen darstellen und über Standortgrenzen hinweg verwendet werden können (Korduan/Zehner).

Eine browserbasierte Web-GIS Anwendung als universelle Lösung mit allen GIS Funktionalitäten ist aber eher unwahrscheinlich. Vielmehr sind bereits bei der Planung die zukünftigen Funktionalitäten festzulegen und anschließend dafür eine passende Architektur, sowie eine Hard- und Software Planung für das System zu erstellen. Denn bei Web-GIS Anwendungen hängen Funktionsumfang und das dahinter stehende System stark voneinander ab.

Im Kontext des erstellten LaFIS-Web Prototypen gilt es auch noch zu überdenken, ob die Performance des Editor Tasks im Produktiveinsatz ausreicht und ob nicht eine Auslagerung der topologischen Beziehungen in die Datenbank sinnvoller wäre. Dadurch müsste ein höherer Implementierungsaufwand für die dazu benötigten User Interfaces und Werkzeuge in Kauf genommen werden.

Denkbar wäre aber auch eine kombinierte Lösung zwischen Web-GIS Client und Desktop Applikation, bei der im Web-GIS Client die Editierfunktionalitäten und die einfachen und wichtigsten topologischen Beziehungen implementiert sind. Die Prüfung von komplizierten und aufwändigen topologischen Fragestellungen wäre dann im Desktop GIS durchzuführen. Mit diesem Ansatz hielte sich eine sehr aufwändige Implementierung in Grenzen. Er bietet aber dennoch die Möglichkeit, einen konsistenten Datenbestand zu erreichen, der alle topologischen Beziehungen berücksichtigt.

6.3 AUSBLICK

Der Editiervorgang im LaFIS-Web Prototypen ist noch nicht ausreichend performant, um eine ausreichende Benutzerfreundlichkeit zu garantieren. ESRI hat mit der Veröffentlichung der ArcGIS Version 9.3.1 eine deutliche Verbesserung der Performance des Editor Tasks angekündigt. Im Rahmen dieser Arbeit konnte dies jedoch noch nicht

getestet werden. Eine Verbesserung der Geschwindigkeit des Editor Tasks würde für den Prototypen eine höhere Benutzerakzeptanz bedeuten. Eine Migration auf die neueste Version dürfte nach Darstellungen von ESRI jedoch keinen größeren Aufwand oder keine Probleme bereiten.

Bisher sind noch nicht alle in 3.1.2 definierten topologischen Beziehungen im neuen Tool „*Add polygon with topology check*“ umgesetzt. Die fehlenden Regeln 2b, 2d und teilweise 2f erfordern noch Lösungen für die Implementierung, lassen jedoch einen höheren Implementierungsaufwand durch die komplexen Fragestellungen vermuten.

Das neu implementierte Editier Tool bietet dem Anwender bisher keine Möglichkeit, Einfluss auf das resultierende Polygon zu nehmen. Der User kann nicht bestimmen, ob die Überlappungen vom neuen Polygon oder von der bestehenden Geometrie entfernt werden. Ein Dialog im Client, der diese Funktionalität bietet, würde die Benutzerfreundlichkeit der Anwendung weiter verbessern.

Die LaFIS-Web Anwendung ist für einen Einsatz in Ämtern und Verwaltungsbehörden gedacht und erfordert dadurch eine Verwendbarkeit für mehrere Benutzer, deren Zahl durchaus 50 bis 100 betragen kann. Dadurch ist eine genaue Planung der Architektur mit den eingesetzten Hard- und Softwarekomponenten notwendig. Um eine solche Anwendung jedoch besser planen zu können, sind noch weitere Performancetests notwendig. In einem produktiven System ist es zudem dringend empfehlenswert, statische Daten zu cachen, um so die Renderzeiten für die Kartendarstellung zu verringern.

LITERATURVERZEICHNIS/BIBLIOGRAPHIE

AGS Help 1, 2009: ESRI, *Components of an ArcGIS Server system*, [online], verfügbar bei: http://webhelp.esri.com/arcgisserver/9.3/java/index.htm#components_of_server.htm
Aufgerufen am 25.06.2009

AGS Help 2, 2009: ESRI, *Working with the ArcGIS Server*, [online], verfügbar bei: http://webhelp.esri.com/arcgisserver/9.3/java/index.htm#working_with_gis_svr.htm
Aufgerufen am 25.06.2009

AGS Help 3, 2009: ESRI, *Working with the ArcGIS Server*, [online], verfügbar bei: http://webhelp.esri.com/arcgisserver/9.3/java/index.htm#what_can_you_publish.htm
Aufgerufen am 25.06.2009

AGS Help 4, 2009: ESRI, *Map authoring considerations for ArcGIS Server*, [online], verfügbar bei: http://webhelp.esri.com/arcgisserver/9.3/java/index.htm#map_authoring.htm
Aufgerufen am 29.06.2009

AGS Help 5, 2009: ESRI, *Introduction to creating Web applications with Manager*, [online], verfügbar bei: http://webhelp.esri.com/arcgisserver/9.3/java/index.htm#intro_web_apps_mgr.htm
Aufgerufen am 29.06.2009

AGS Resource Center 1, 2009: ESRI, *Customize the EditorTask*, [online], verfügbar bei: http://resources.esri.com/help/9.3/arcgisserver/adf/java/help/concepts_start.htm#doc/517d09a3-b83d-484e-9d9f-e2e2b32eaa8f.htm
Aufgerufen am: 09.07.2009

ArcGIS Desktop 1, 2009: ESRI, *An overview of the geodatabase*, [online], verfügbar bei: http://webhelp.esri.com/arcgisdesktop/9.3/index.cfm?TopicName=An_overview_of_the_geodatabase
Aufgrufen am 03.06.2009

ArcGIS Desktop 2, 2009: ESRI, *Understanding versioning*, [online], verfügbar bei: http://webhelp.esri.com/arcgisdesktop/9.3/index.cfm?TopicName=Understanding_versioning
Aufgrufen am 03.06.2009

ArcGIS Server, 2009: ESRI, ArcGIS *Server Product Life Cycle Support Status*, [online], verfügbar bei
http://downloads2.esri.com/support/product%20life%20cycle/other_/ArcGISServer93_PLC.pdf
Aufgerufen am: 20.06.2009

Asche H., 2001: *Kartographische Informationsverarbeitung in Datennetzen*, erschienen in: Hermann/Asche: *Web.Mapping 1*, Heidelberg, Herbert Wichmann Verlag

Barthelme N., 2005: *Geoinformatik - Modelle, Strukturen, Funktionen*; Berlin, Springer Verlag

Bengel G., 2004: *Grundkurs Verteilte Systeme*, Wiesbaden, Vieweg Verlag, 3. Auflage

Bill R., 1999: *Grundlagen der Geo-Informationssysteme*, Band 1: *Hardware, Software und Daten*, Heidelberg, Herbert Wichmann Verlag

ESRI D 1, 2009: ESRI Geoinformatik GmbH, *ArcGIS Server*, [online], verfügbar bei: http://www.esri-germany.de/products/arcgis/arcgisserver/index.html
Aufgerufen am: 05.06.2009

ESRI Press, 2004: *ArcGIS 9 - ArcGIS Server Administrator and Developer Guide*, verschiedene Autoren, Verlag: ESRI Press

ESRI Support Center 1, 2009: *ArcGIS Server Supported Webbrowsers*, [online], verfügbar bei:
http://wikis.esri.com/wiki/display/ag93bsr/ArcGIS+Server+Supported+Web+Browsers
Aufgerufen am 20.08.09

Farley J., / Crawford W., 2005: *Java Enterprise in a Nutshell – A Practical Guide*, Sebastopol CA, O`Reilly Verlag, 3. Auflage

Korduan P. / Zehner M., 2008: *Geoinformation im Internet*, Heidelberg, Wichmann Verlag

Krause A., 2006: *Einführung eines GIS für die Landwirtschaftsverwaltung der BRD auf Grundlage EU-rechtlicher und nationaler Verordnungen*, Göttingen: Georg-August-Universität, Herausgeber: Prof. Dr. Martin Kappas, ibidem-Verlag Stuttgart

Krüger G., 2006: *Handbuch der Java-Programmierung*, München, Addison-Wesley Verlag, 4.Auflage

SELFHTML 1, 2009: SELFHTML: *Einführung / Web Technologien / HTML*, [online], verfügbar bei: http://de.selfhtml.org/intro/technologien/html.htm
Aufgerufen am. 16.06.2009

SELFHTML 2, 2009: SELFHTML: *Einführung / Web Technologien / Stylesheets*, [online], verfügbar bei: http://de.selfhtml.org/intro/technologien/css.htm
Aufgerufen am. 16.06.2009

SELFHTML 3, 2009: SELFHTML: Einführung / *Web Technologien / Einführung in XML*, [online], verfügbar bei: http://de.selfhtml.org/xml/intro.htm#w3c
Aufgerufen am. 16.06.2009

SELFHTML 4, 2009: SELFHTML: *Einführung / Web Technologien / Einführung in Java Script und DOM*, [online], verfügbar bei: http://de.selfhtml.org/javascript/intro.htm
Aufgerufen am. 16.06.2009

SELFHTML 5, 2009: SELFHTML: *Einführung / Web Technologien / Allgemeines zu dynamische HTML*, [online], verfügbar bei: http://de.selfhtml.org/dhtml/intro.htm
Aufgerufen am. 16.06.2009

Strobl J., 2001: *Online-GIS – das WWW als GIS-Plattform*, erschienen in: Hermann/Asche: *Web.Mapping 1*, Heidelberg, Herbert Wichmann Verlag

Wikipedia 1, 2009: Wikipedia, *Hypertext Transfer Protocol*, [online], verfügbar bei: http://de.wikipedia.org/wiki/Hypertext_Transfer_Protocol
Aufgerufen am: 16.02.2009

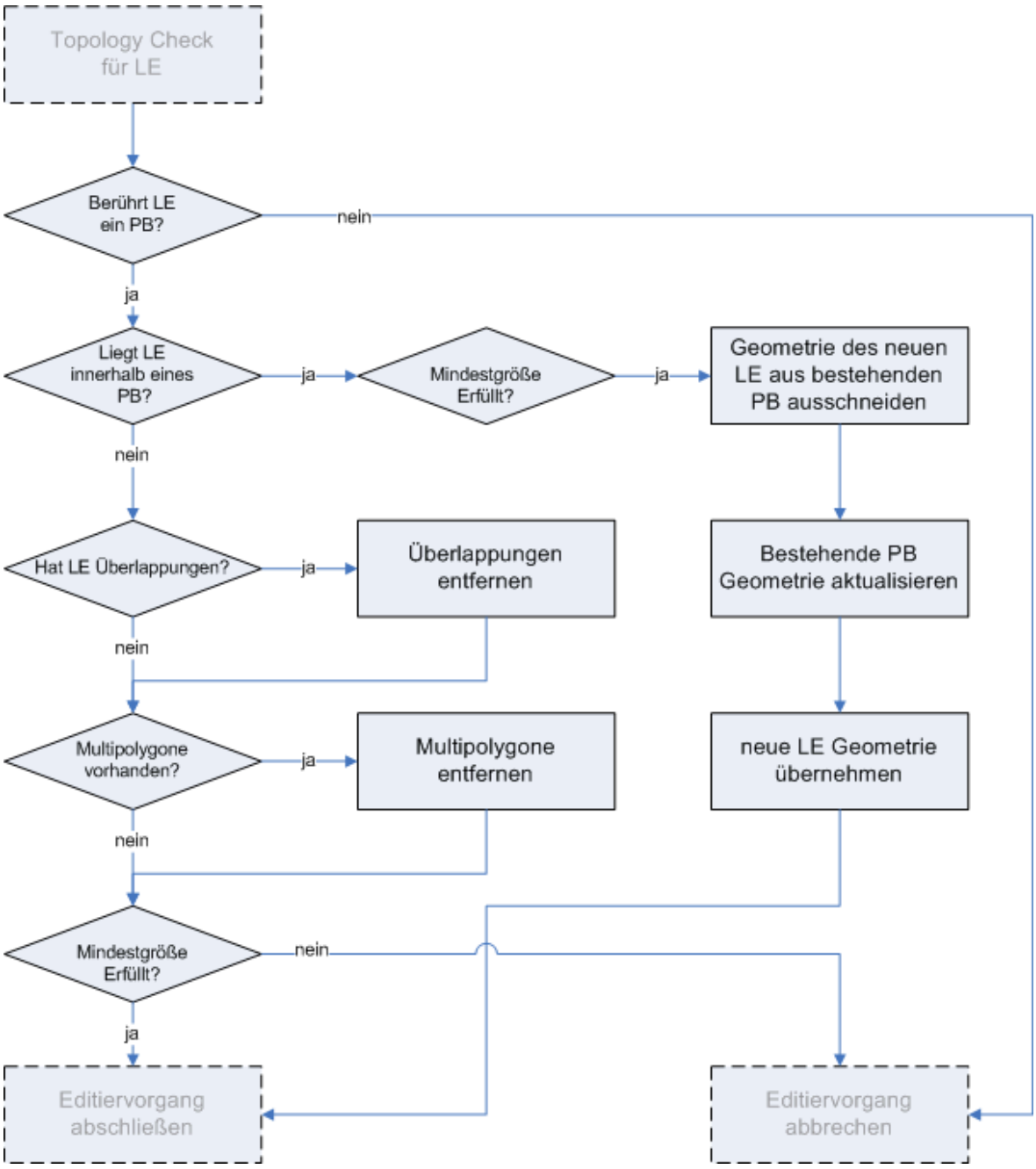
Wikipedia 2, 2009: Wikipedia, *Web Map Service*, [online], verfügbar bei: http://de.wikipedia.org/wiki/Web_Map_Service
Aufgerufen am: 16.02.2009

Wikipedia 3, 2009: Wikipedia, *Web Feature Service*, [online], verfügbar bei: http://de.wikipedia.org/wiki/Web_Feature_Service
Aufgerufen am: 16.02.2009

Wikipedia EN 1, 2009: Wikipedia, *ArcGIS – Wikipedia*, [online], verfügbar bei: http://en.wikipedia.org/wiki/ArcGIS
Aufgerufen am: 05.02.2009

ANLAGEN

ANLAGE 1: ABLAUFDIAGRAMM LETOPOCHECK



ANLAGE 2: ABLAUFDIAGRAMM NEATOPOCHECK