Inhaltsverzeichnis

1 GRUNDLAGEN 8

1.1 Geographische Informationssysteme 8

1.1.1 WebGIS 8

2 GRÜNDE FÜR EINE „PRINT ON DEMAND“ FUNKTION 10

2.1 Recherche im Internet 10

2.2 Fazit 12

3 ENTWICKLUNGSUMGEBUNG 13

3.1 Das Microsoft .NET Framework 13

3.1.1 Beschreibung 13

3.1.2 Die Programmiersprache C# 13

3.2 Application Developer Framework (ADF) des ArcGIS 9 Servers 14

3.2.1 Überblick 14

3.3 ArcGIS Server 9.0 der Firma ESRI 15

3.3.1 Beschreibung 15

3.3.2 Die ArcGIS Serverarchitektur 16

3.3.3 Das ArcGIS Server System 17

3.3.3.1 Überblick 17

3.3.3.2 Der Server Object Manager 17

3.3.3.3 Der Server Object Container 18

3.3.4 ArcCatalog (ArcGIS Desktop) 18

3.3.4.1 Überblick 18

3.3.5 Die GIS Server Sicherheit 20

3.3.5.1 Überblick 20

3.3.5.1.1 Sicherheit innerhalb des GIS Server 20

3.3.5.1.2 Sicherheit innerhalb einer *Web Application* oder *Web Service* (*Impersonation*) 21

3.3.6 Der Vergleich zu ArcIMS 21

3.4 Arbeitsumgebung 22

3.4.1 Hardware 22

4 BESCHREIBUNG DER WICHTIGSTEN OBJEKT MODELL DIAGRAMME 23

4.1 Beschreibung 23

4.2 Das Display Object Model Diagram 24

4.2.1 Überblick 24

4.2.2 Beschreibung der relevanten Schnittstellen 26

4.2.2.1 Das IDisplay Interface 26

4.2.2.2 Das IScreenDisplay Interface 28

4.2.2.3 Das IDisplayTransformation Interface 29

4.2.2.3.1 Die *tagRECT* Struktur der *DeviceFrame* Eigenschaft 30

4.3 Das Geometry Object Model Diagram 30

4.3.1 Überblick 30

4.3.2 Beschreibung der relevanten Schnittstellen 31

4.3.2.1 Das IEnvelope Interface 31

4.4 Carto Object Model Diagram 34

4.4.1 Überblick 34

4.4.2 Beschreibung der relevanten Schnittstellen 35

4.4.2.1 Das IActiveView Interface 35

4.4.2.1.1 Die *ExportFrame* Eigenschaft 36

4.4.2.1.2 Die Output() Methode 37

4.4.2.2 Das IPageLayout Interface 37

4.4.2.3 Das IPage Interface 39

4.4.2.3.1 Die *FormID* Eigenschaft 40

4.4.2.3.2 Die *Orientation* Eigenschaft 40

4.4.2.3.3 Die *Units* Eigenschaft 40

4.4.2.3.4 Die *QuerySize()* Methode 41

4.5 Die Output Bibliothek 41

4.5.1 Überblick 41

4.5.2 Beschreibung der relevanten Schnittstellen 43

4.5.2.1 Das IExport Interface 43

4.6 Die ESRI.ArcGIS.Server Bibliothek 44

4.6.1 Überblick 44
4.6.2 Das IServerContext Interface 44

4.7 Der ESRI.ArcGIS.Server.WebControls Namespace 45
4.7.1 Die *WebPageLayout* Klasse 45
4.7.1.1 Die *Export()* Methoden 47
4.7.1.1.1 Der Export mit der *ESRI.ArcGIS.Carto.esriImageFormat* Enumeration 47
4.7.1.2 Der *WebPageLayout()* Konstruktor 48

4.8 Der System.Web.SessionState Namespace 48
4.8.1 Beschreibung 48
4.8.1.1 Die *HttpSessionState* Klasse 49

4.9 Der System.Web.Mail Namespace 49
4.9.1 Beschreibung 49
4.9.1.1 Die *MailMessage* Klasse 50
4.9.1.2 Die *SmtpMail* Klasse 50

5 PLANUNG UND ENTWICKLUNG DER EINZELNEN „*PRINT ON DEMAND*" KOMPONENTEN 52

5.1 Die Planung 52
5.1.1 Einarbeitung in die Programmiersprachen Visual C#, VB .NET, ASP .NET und XML. 52
5.1.1.1 Einarbeitung in die Objekt Modelle und ArcObjects Programmierung. 54
5.1.1.2 Einarbeitung in die *ArcObjects* Programmierung. 56
5.1.2 Test von kleinen Code-Beispielen der ArcGIS Developer Help 57
5.1.2.1 Test der Export-Funktion im ArcMap 58

5.2 Realisierung und Testen der Funktionen 60
5.2.1 Das PageLayoutViewer Template des ADF .NET Frameworks 60
5.2.1.1 Einbindung des WebUserControls (Web-Benutzersteuerelement) 61
5.2.1.2 Entwicklung der Funktionen „Export Format" und „Auflösung" 61
5.2.1.2.1 Ergebnis 63
5.2.1.3 Entwicklung der Funktion „E-Mail Adresse" 65
5.2.1.3.1 Ergebnis 66
5.2.1.4 Die Konfiguration der XML Datei 66

5.3 Besonderheiten und Probleme 67

5.3.1 Einbindung des Impersonation 67
5.3.2 Literatur und fehlerhafte Codes 67

6 AUSBLICK 68

6.1 Konzept 68
6.1.1 Entwicklung der Komponente „Papier-Format" 69
6.1.1.1 Beschreibung 69
6.1.2 Entwicklung der Komponente „Maßstab" 69
6.1.2.1 Beschreibung 69
6.1.2.1.1 Das *IMapDescription* Interface der Carto Bibliothek 70
6.1.2.1.2 Das *IMapArea* Interface der Carto Bibliothek 71
6.1.2.1.3 Das *IPoint* Interface der Geometry Bibliothek 72
6.1.2.1.4 Das *ICenterAndScale* Interface der Carto Bibliothek 73
6.1.2.1.5 Die Methode im Überblick 74
6.1.3 Die Generierung der Legende 74
6.1.3.1 Überblick über die Programmierung 74
6.1.3.1.1 Das *IMapFrame* Interface der Carto Bibliothek 75
6.1.3.1.2 Das *IMapSurroundFrame* Interface der Carto Bibliothek 75
6.1.3.1.3 Das *ILegend* Interface der Carto Bibliothek 75
6.1.3.1.4 Das *ILegendFormat* Interface der Carto Bibliothek 75
6.1.3.1.5 Das IElement Interface der Carto Bibliothek 75
6.1.3.2 Möglicher Ablauf der Generierung einer Legende 75
6.1.4 Einbindung eines Webservices 76

7 ZUSAMMENFASSUNG 77

8 TABELLENVERZEICHNIS 78

9 LITERATUR 79

10 LITERATUR ONLINE 80

11 ABKÜRZUNGSVERZEICHNIS 81

Abbildungsverzeichnis

Abb. 1: .WebGIS Architektur 8

Abb. 2: .Beispiel Export Online Bayern 11

Abb. 4: .NET Framework Hirarchie 13

Abb. 5: .NET ADF Framework Hirarchie 14

Abb. 6: ArcGIS Serverarchitektur 16

Abb. 7: Systemarchitektur 17

Abb. 8: ArcCatalog 18

Abb. 9: MapServer Objekt Modell 19

Abb. 10: Sicherheitskomponenten des ArcGIS Server 20

Abb. 11: ESRI Objekt Modell 23

Abb. 12: Das Display Objekt Modell 25

Abb. 13: IDisplay Objekt Modell ESRI 26

Abb. 14: IScreen Objekt Modell ESRI 28

Abb. 15: IEnvelope Objekt Modell ESRI 31

Abb. 16: Envelope Objekt 32

Abb. 17: Die PutCoords() Methode 33

Abb. 18: Die IActiveView Struktur 35

Abb. 19: Die IActiveView Objekt Modell 36

Abb. 20: Das PageLayout Objekt Modell 37

Abb. 21: Das IPage Objekt Modell 39

Abb. 22: Das IExport Objekt Modell 43

Abb. 23: Das Objekt Modell der WebPageLayout Klasse 45

Abb. 24: Code-Ausschnitt der Format Methode 47

Abb. 25: Code-Ausschnitt der Mail Methode 50

Abb. 26: Hauptfenster ArcGIS Developer Help 54

Abb. 27: Objekt Modell der ArcGIS Developer Help 55

Abb. 28: Anleitung Entwickeln von Objekten der ArcGIS Developer Help 56

Abb. 29: Anleitung Entwickeln von Objekten der ArcGIS Developer Help 57

Abb. 30: Export-Funktion ArcMap, BoundingBox 58

Abb. 31: Export-Funktion ArcMap, Dialog-Fenster 58

Abb. 32: Ansicht im Acrobat Reader 59

Abb. 33: PageLayoutViewer Template 60

Abb. 34: Auflösung und Format des UserControls 61

Abb. 35: Code Export-Funktion 62

Abb. 36: PageLayoutViewer mit Exportelement 63

Abb. 37: Datei-Verzeichnis der Ausgabe-Datei 64

Abb. 38: E-Mail Komponente des UserControls 65

Abb. 39: Code Sende Methode 65

Abb. 40: E-Mail Account mit Nachricht 66

Abb. 41: XML Code Impersonation 66

Abb. 42: XML Code SessionState 67

Abb. 43: MapViewer mit Layoutelement 68

Abb. 44: UserControl Layout 69

Abb. 45: IMapDescription Interface 70

Abb. 46: IMapArea Interface 71

Abb. 47: IPoint Interface 72

Abb. 48: ICenterAndScale Interface 73

Abb. 49: WebService 76

Einleitung

Das Internet wird täglich von mehreren Millionen Menschen benutzt. Durch die Nutzung von innovativer Internet-Technologien ist die Entwicklung der Geographischen Informationssysteme für das WWW (WebGIS) heute so fortgeschritten, das sie jedem Benutzer die Möglichkeit bietet, ohne spezielle Software, Kartenmaterial zu betrachten, zu analysieren und zu bearbeiten.

Es genügt ein einfacher Webbrowser, um Karten in unterschiedliche Datenformate anzuzeigen. Es ist nicht zu übersehen, dass sich eine Vielzahl von Firmen auf dieses Gebiet spezialisiert haben. Dies könnte sich aber auch nachteilig auswirken, da die Übersichtlichkeit darunter leiden könnte.

Das Informationstechnische Zentrum (ITZ) der Landesanstalt für Umweltschutz (LfU) in Karlsruhe ist für die Erfassung, Speicherung und Bereitstellung sämtlicher umweltrelevanter und Geobezogener Daten innerhalb Baden-Württembergs verantwortlich. Außerdem wird die Entwicklung der Fachanwendungen der LfU und der Umweltdienststellen im ganzen Land unterstützt.

Um die Umweltdaten schnell und aktuell auszuliefern, werden diese unter anderem per Internet graphisch in einem Kartendienst dargestellt. Die Verbreitung der geographischen Daten auf diese Weise hat den Vorteil, dass die Benutzer, die überwiegend aus dem öffentlichen und gewerblichen Bereich stammen, keine besondere Software einsetzen müssen, da zum Betrachten ein normaler Browser ausreicht. Es bestehen bereits mehrere Kartendienste zu den verschiedenen Themen, wie z.B. Luft, Wasser, Abfall oder Naturschutzgebiete.

Die Kartendienste werden zur Zeit auf Basis von HTML-Viewern und der Programmiersprache Javascript bereit gestellt. Desweiteren benutzt die LfU für ihre Kartendienste den ArcIMS-Server der Firma ESRI, der nur bedingt den Bedürfnissen der Benutzer eines solchen Dienstes gerecht wird.

Um diese Kartendienste flexibler gestalten zu können, ist es erforderlich, den Benutzern eines solchen Dienstes eine qualitativ hochwertige Export –und Druckmöglichkeit zu bieten.

Das Ziel der vorliegenden Diplomarbeit ist nun, eine Export –und Druckfunktion zu entwickeln, welche den Benutzern die Wahl zwischen den verschiedenen Papierformaten, Maßstäbe und Auflösungen zur Verfügung stellt. Diese Funktion wird mit der neuen Programmiersprache Visual C#, und auf Basis der neuen Software ArcGIS 9 Server, entwickelt.

Im Zuge dieser Diplomarbeit wird sich aber herausstellen, dass der ArcGIS 9 Server, welcher von der Firma ESRI entwickelt wurde, noch Mängel in der Umsetzung der Programmierung aufweist. Deshalb konnten nicht alle Funktionen zufrieden stellend entwickelt werden.

1 Grundlagen

1.1 Geographische Informationssysteme

1.1.1 WebGIS

GIS-Anwendungen im Inter- oder Intranet, kurz WebGIS, machen Informationen für eine große Zahl von Anwendern zugänglich, sind plattformunabhängig und erfordern keine Installation und kostenintensiver Desktop-GIS-Software. Zur Anzeige der GIS-Anwendungen ist lediglich ein Internet-Browser erforderlich. Schon diese Aspekte verdeutlichen das grosse Potential von WebGIS-Anwendungen. Für viele Nutzer, besonders auch in öffentlichen Verwaltungen, ist Kartenmaterial als Auskunftssystem sehr interessant. Mit zunehmender Geschwindigkeit und Entwicklung der Technik, wird das Auslagern von GIS-Funktionalität in den normalen Webbrowser ohne lokale Installation immer vorteilhafter.
WebGIS bedeutet, dass der Nutzer mindestens eine dynamische Karte sieht, sich in dieser frei bewegen kann und diese selbst, aufgrund einer bestimmten Auswahl an Themen, frei gestalten kann. Dies bedeutet generell, dass eine serverseitige Architektur aufgebaut werden muss. Diese sieht im einfachsten Fall so aus, dass ein Webserver, und ein Kartenserver (MapServer), aufgesetzt werden und ihr Zusammenspiel erlaubt, dass der Nutzer an einer Benutzeroberfläche, dem Client, navigiert, diese Anfrage an den MapServer geschickt wird, welcher eine neue Karte produziert und an den Client zurückschickt.[1]
Die folgende Abbildung soll dies noch einmal verdeutlichen:

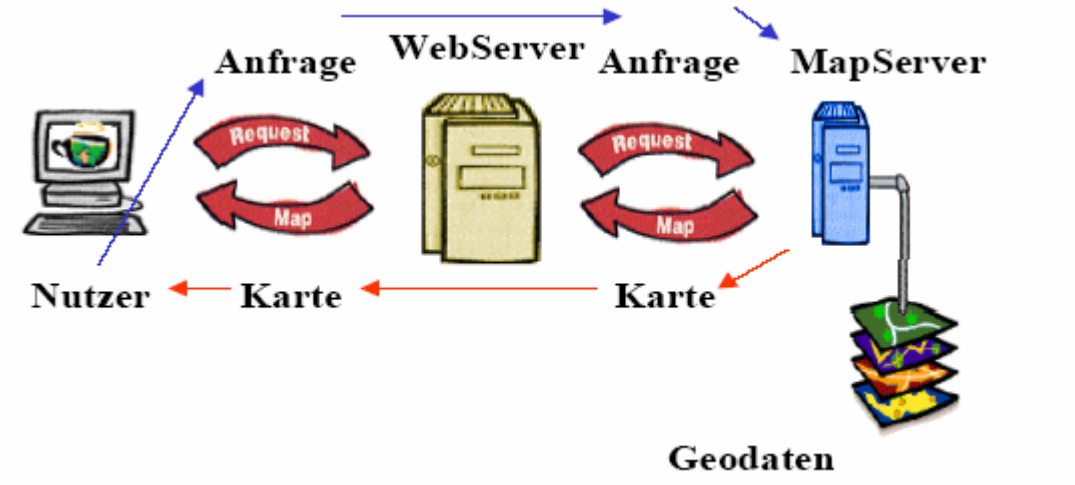

Abb. 1: .WebGIS Architektur

[1] Vgl.: Praxishandbuch_WebGIS, Seite 41

Ein WebGIS basiert auf dem Client/Server Prinzip, da der Zugriff von einem dezentralen Rechner des Anwenders (Client) zu dem Server erfolgt, welcher eine Verbindung zu einer Geodatenbank besitzt. Er beinhaltet noch zusätzlich Funktionen, die das Analysieren und Präsentieren dieser Daten ermöglicht.
Da der Client auch je nach Umfang verschiedene Funktionen ausführen kann, spricht man jeweils von einem:

- ***Thin Client***
- ***Thick Client***
- ***Medium Client***

Der ***Thin Client*** erstellt nur Anfragen an den Server, der die Daten dann analysiert und aufbereitet.
Der ***Thick Client*** fordert Daten vom Server an, die er dann selber bearbeiten kann.
Der ***Medium Client*** verarbeitet einen Teil der Daten. Der andere Teil wird von dem Server verarbeitet.[2]

Ein WebGIS ist auf mehreren Schichten aufgebaut. Diese Schichten werden unterteilt in:

- **Verarbeitungsschicht**
- **Datenschicht**
- **Präsentationsschicht**

In der **Verarbeitungsschicht** nimmt der Web Server Anfragen des Clients entgegen. Der Web Server erstellt statische Webseiten und schickt die Anfrage an den Applikation Server. Dieser prüft und verarbeitet die Anfragen. Bei fehlenden Angaben wird die Anfrage unter Umständen an den Web Server mit der Forderung nach zusätzlichen Informationen zurückgeschickt. Die Anfrage wird dann weiter an den Map Server geschickt, welcher diese Anfragen verarbeitet und als Ergebnis z.B. eine Karte erzeugt.
In der **Datenschicht** liefert der Daten Server die erforderlichen Daten an den Map Server.
In der **Präsentationsschicht** werden die verarbeitenden Daten wieder vom Web Server an den Client geschickt.[3]

[2] Vgl.: Vorlesungsskript, GIS II
[3] Vgl.: http://www.sogi.ch/sogi/Technologie2.pdf, Seite 13

2 Gründe für eine „Print on demand" Funktion

2.1 Recherche im Internet

Das Ziel einer Diplomarbeit ist sicherlich einerseits etwas Neues zu erschaffen oder andererseits altes weiter zu entwickeln. Dies beinhaltet in beiden Fällen, zunächst eine intensive Recherche von Literatur und anderen Quellen vorzunehmen. Da dieses Projekt als Zielrichtung das Internet und Intranet vorgibt, ist es wichtig einige Vergleiche in diesem Bereich anzustellen.
Die Strukturierung der Recherche setzt sich aus den folgenden Punkten zusammen:

1. Wer stellt Kartendienste bereit
2. Welche Software wird dafür eingesetzt
3. Welche Funktionalitäten besitzen diese Kartendienste
4. Wie ist die Qualität der Ergebnisse (bezüglich des Druckformates)

Zu Punkt 1):

Da die Entwicklung im Bereich der Kartendienste stetig wächst, spezialisieren sich verschiedene Firmen und Behörden darauf, Kartenwerke via Internet zu veröffentlichen. Aufgrund der zentralen Verfügbarkeit von Geodaten, werden diese Dienste vermehrt in Umwelt –und Geologischen Ämtern eingesetzt. Auch private Firmen nutzen diesen Bereich, um der breiten Öffentlichkeit Informationen, z.B. Routenplaner, zu vermitteln.

Zu Punkt 2):

Der UMN (University of Minnesota) MapServer ist einer der bekanntesten OpenSource Server, der über eine vollständige Plattformunabhängigkeit verfügt. Mindestens genauso bekannt und häufig eingesetzt ist der ArcIMS Server der Firma ESRI.

Zu Punkt 3):

Die meisten Kartendienste beinhalten die üblichen Funktionen einer Navigation (z.B. Verschieben, Zoomen usw.). Weiterhin besitzen diese Dienste nur bedingte Möglichkeiten, die Einstellungen bzgl. der Druckausgabe zu definieren (z.B. Auflösung, Papierformat).
Einen zusätzlichen Service, wie z.B. einen E-Mail Versand, wurde bei keinem aktuellen Kartendienst bereitgestellt.

Zu Punkt 4):

Die Druckformate der meisten Kartendienste werden als JPEG oder PNG dargestellt.

Das folgende Beispiel zeigt exemplarisch den GeoFachdatenAtlas des Geologischen Bayrischen Landesamtes:

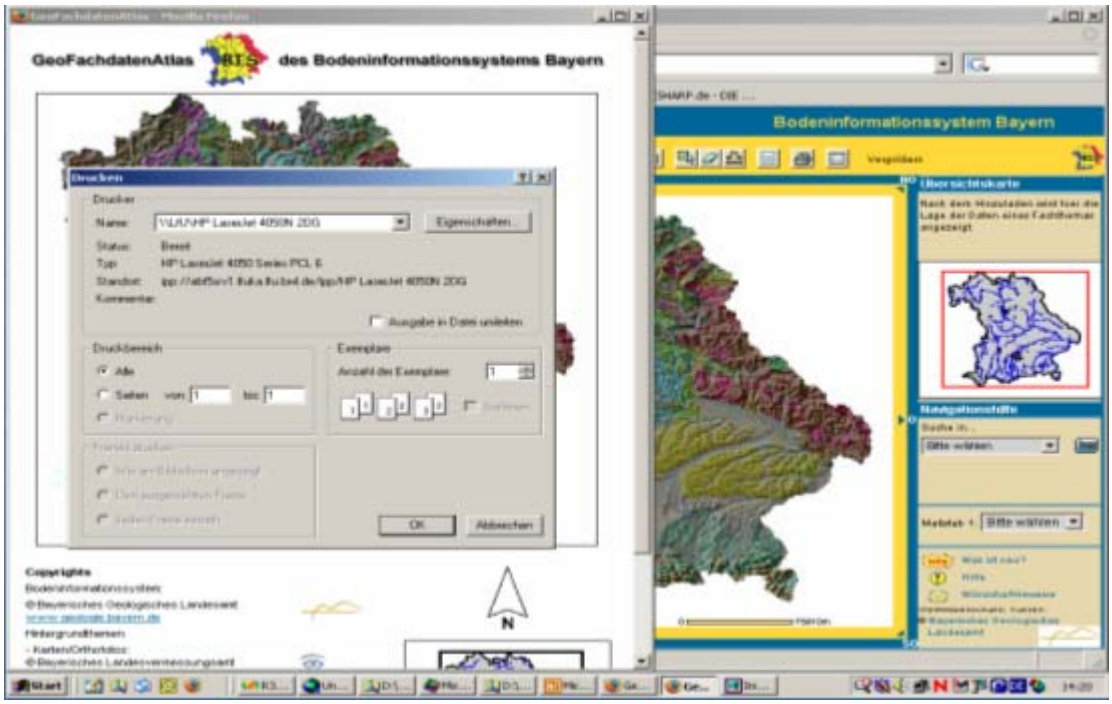

Abb. 2: .Beispiel Export Online Bayern

Quelle: http://www.bis.bayern.de/bis/index.html

Bei diesem Online-Atlas kann man nach der Wahl eines Ausschnittes einen Print-Button betätigen. Daraufhin öffnet sich ein PopUp-Fenster mit der Layoutansicht des ausgewählten Ausschnittes. Die Karte wird im JPEG-Format angezeigt. Danach kann man durch das Drücken eines Print-Buttons, anhand des Microsoft Druck-Menü, das Layout ausdrucken.
Nachteilig ist das Format und die fehlende Möglichkeit, die Auflösung auszuwählen.

2.2 Fazit

Die Recherche im Internet hat aufgezeigt, dass im Bereich der benutzerdefinierten Export – und Druckmöglichkeiten innerhalb eines Kartendienstes der Entwicklungsspielraum noch nicht voll ausgeschöpft ist.

Das Ziel der Diplomarbeit ist, eine benutzerdefinierte Export- und Druckfunktion zu entwickeln, die dem Anwender eine qualitativ hochwertige Druckmöglichkeit via Internet oder Intranet bietet. Der Anwender soll die Datei per E-Mail abrufen können und sich diese dann ausdrucken.

Anhand dieses Zieles, kann man davon ausgehen, dass eine Weiterentwicklung in diesem Bereich sicherlich gerechtfertigt ist.

3 Entwicklungsumgebung

3.1 Das Microsoft .NET Framework

3.1.1 Beschreibung

Das .NET Framework ist eine neue Programmierplattform, die Microsoft im Jahr 2000 für das eigene Betriebssystem entwickelt hat. Das Framework beinhaltet weiterhin den Kommandozeilencompiler für die Sprachen Visual Basic.NET, C# und JScript.NET, so dass dies als Grundpaket für die Entwicklung von. NET-Anwendungen bereits ausreicht.
Da jede Sprache eine eigene Laufzeitumgebung voraussetzt, kommt hier die Common Language Runtime (*CLR*) zu tragen, die zur Programmierungskompabilität beiträgt.
Das Framework beinhaltet weiterhin eine einheitliche Klassenbibliothek, die für objektorientierte Sprachen entwickelt wurde. Dies hat den Vorteil, dass jede der zur Verfügung stehenden Sprachen auf die gleichen Klassen zugreifen kann.[4]
Die folgende Abbildung zeigt die zentralen Bestandteile des .NET Frameworks:

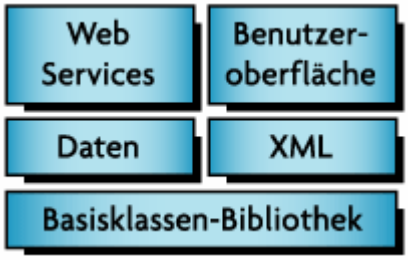

Abb. 4: .NET Framework Hirarchie

Quelle: http://www.elektroniknet.de/topics/automatisieren/fachthemen/2002/0031

3.1.2 Die Programmiersprache C#

C# ist eine neue Programmiersprache, die das .NET Framework bereitstellt. Diese Sprache beinhaltet alle Aspekte der objektorientierten Programmierung. Sie hat den Vorteil, dass sie relativ schnell zu erlernen ist, im Gegensatz zu JAVA oder C++. Aufgrund der Historie der objektorientierten Programmierung kann man behaupten, dass Java von C++, und C# von Java gelernt hat.
C# stellt somit eine gelungene Mischung der Programmiersprachen JAVA und C++ dar.

[4] Vgl.: http://www.dotnetframework.de/default2.aspx

3.2 Application Developer Framework (ADF) des ArcGIS 9 Servers

3.2.1 Überblick

Der ArcGIS 9 Server beinhaltet *ein Application Developer Framework* (ADF), welches auf das .NET Framework aufgesetzt ist. Dies erlaubt die Integration von GIS Funktionalitäten in die Web Steuerelemente. Das *ADF* beinhaltet eine Anzahl von Benutzersteuerelementen und sogenannten Templates (Vorlagen), welche zum Entwickeln von Web Applikationen bestimmt sind.
Durch das Aufsetzen auf das .NET Framework erweitert sich die Klassenbibliothek, mit einer größeren Auswahl an neuen Klassen, erheblich.[5]
Das *.NET ADF* besitzt folgende unterschiedliche Objekte:

- Visual Studio .NET Templates
- Benutzerdefinierte ADF Web Steuerelemente
- Flexible Klassen, um Applikationen für den GIS Server zu entwickeln

Die folgende Abbildung zeigt deutlich die Hirarchie des .NET ADF Framework in der Entwicklungsumgebung:

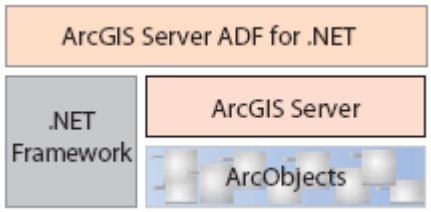

Abb. 5: .NET ADF Framework Hirarchie

Quelle: ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch5.pdf, S. 2

[5] Vgl.: ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch5.pdf, Seite 1

3.3 ArcGIS Server 9.0 der Firma ESRI

3.3.1 Beschreibung

Mit dem ArcGIS Server steht erstmalig ein GIS Server mit voller Geoverarbeitungsfunktionalität zur Verfügung. GIS Funktionalität kann damit in angepassten Lösungen von zentralen Servern bereitgestellt und von verteilten Anwendern genutzt werden. Vor allem für unternehmensweite GIS Infrastrukturen ergänzt ArcGIS Server die bestehende Produktfamilie von Desktop Anwendungen und Daten-/Internet Map Servern um serverseitige GIS Funktionalität, die über Visualisierung hinausgeht.
Mit ArcGIS Server lassen sich Fachanwendungen aus Ver- und Entsorgung, Kataster und Vermessung, behördlicher Planung, Geomarketing oder Umweltsimulation serverseitig abbilden und mit nicht-GIS Verfahren auf Basis von IT-Standards und Web Protokollen koppeln. Dank einer multiplen Plattformkompabilität für .NET und Java lässt sich ArcGIS Server in IT-Infrastrukturen, Datenbankumgebungen und Applikationsserver einfach integrieren. Folgend kann man Funktionen entwickeln, die auf den Betriebssystemen von Windows und UNIX einsetzbar sind.
Durch diese besonders skalierbare Lösung entstehen neue Architekturmöglichkeiten, die einen rascheren „return on investment“ und geringere „total cost of ownership“ (TOC) ermöglichen. Diese Effekte werden durch die zentrale Installation und Wartung und die hohe Integrationsfähigkeit von ArcGIS Server in bestehende IT-Landschaften gefördert. Auch hier zahlt sich die Plattformoffenheit und Modularität der zugrunde liegenden Architektur aus. [6]
Die folgenden Punkte sollen die Funktionen des ArcGIS Servers zusammenfassend darstellen:

- Stellt über den Browser Zugang zu GIS her
- Liefert erweiterten Service über das WebGIS
- Entwicklung kundenspezifischer Anwendungen mit .NET und JAVA
- Integration von GIS und IT-Technologien
- Stellt zentral verwaltete Multiuser Entwicklungen her
- Führt räumliche Analyseoperationen auf dem Server durch

[6] Vgl. http://www.esri-germany.de/products/arcgis/index.html, DATUM

3.3.2 Die ArcGIS Serverarchitektur

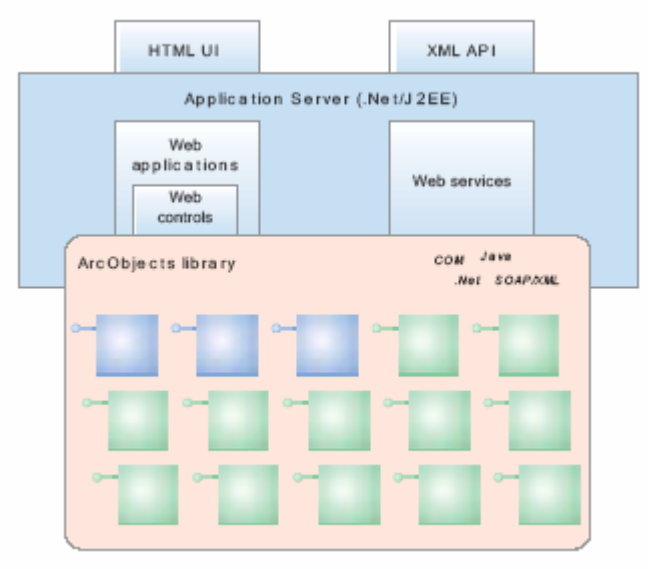

Abb. 6: ArcGIS Serverarchitektur

Quelle: ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 33

Der ArcGIS Server ist grundsätzlich ein Objekt-Server, der eine Menge von GIS Server Objekten verwaltet. Er bietet ein Rahmengerüst (Framework) für Webanwendungen (*Web Applications*) und *Web Services*.
Anwendungsentwickler im ArcGIS Bereich arbeiten speziell mit diesen Serverobjekten, die auch *ArcObjects* genannt werden. *ArcObjects* besitzen mehrere Entwicklungsinterfaces (APIs), die COM, .NET, Java und C++ beinhalten. Eine der Vorteile von *ArcObjects* ist die umfangreiche Bibliothek, in der Anwender mit dem Server über den Webbrowser interagieren können. [7]

[7] Vgl.: ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf,

3.3.3 Das ArcGIS Server System

3.3.3.1 Überblick

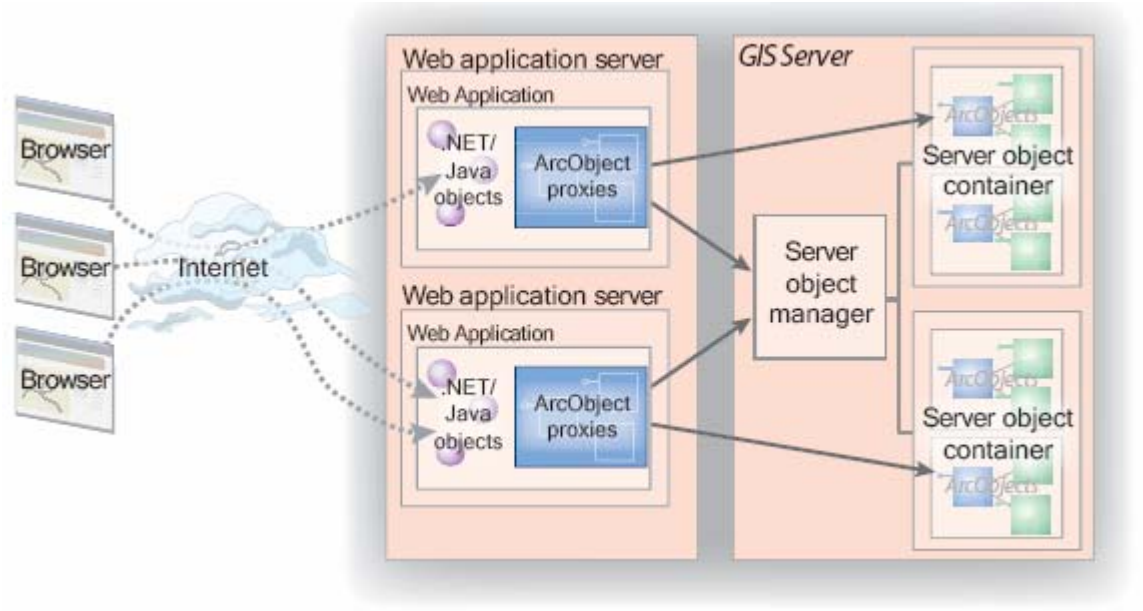

Abb. 7: Systemarchitektur

Quelle: ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S.35

Bei der Kommunikation des Browsers eines Benutzers mit dem *Web Application Server*, wird das ArcGIS Server *Application Developer Framework* angesprochen. Dieses beinhaltet die .NET und Java Entwicklungsumgebung.
Die *Web Application* benutzt den Proxy, um mit den *ArcObjects* zu arbeiten. Die gesamte Durchführung geschieht jedoch im GIS Server. Die *Web Application* sendet wiederum eine Anfrage an den *Server Object Manager*, der mit Hilfe des Proxy die *Server Objects* im Gis Server benutzt.

3.3.3.2 Der Server Object Manager

Der GIS Server besteht aus einem *Server Object Manager* (SOM), welcher ein Windows Service ist. Dieser läuft auf einer einzelnen Maschine. Der *Server Object Manager* verwaltet einen oder mehrere *Server Object Con*tainer (SOC). Wird von einer Anwendung eine Connection über LAN oder WAN zum GIS Server hergestellt, spricht diese den Namen oder die IP Adresse des *Server Object Manager* an.

3.3.3.3 Der Server Object Container

Der *Server Object Container* beinhaltet die Serverobjekte, die vom *Server Object Manager* verwaltet werden. Diese Objekte sind *ArcObjects*-Komponenten oder Map Objekte, die gleichmäßig auf *allen Server Object Container* verteilt sind. Sie werden bei der Installation als Teil des ArcGIS Server Systems integriert.

3.3.4 ArcCatalog (ArcGIS Desktop)

3.3.4.1 Überblick

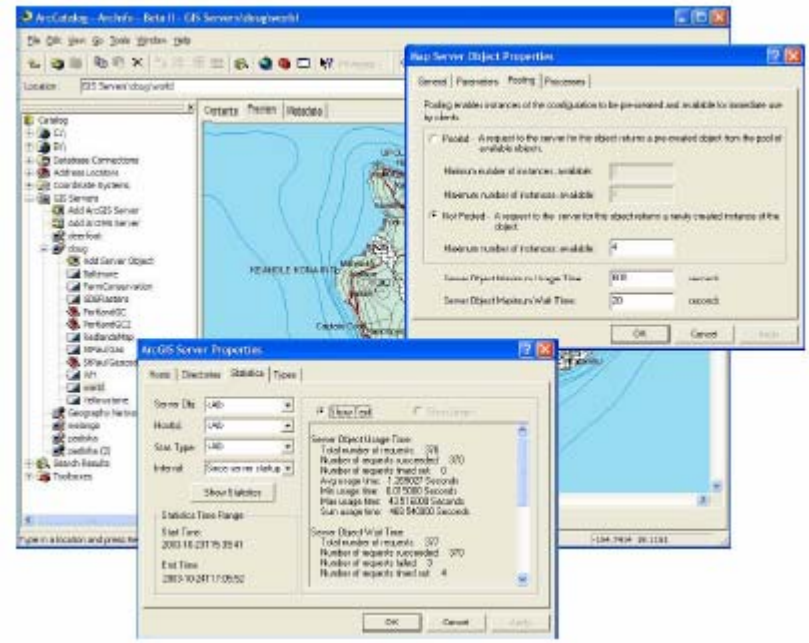

Abb. 8: ArcCatalog

Quelle: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 39

ArcCatalog ist eine ArcGIS Desktop Applikation, durch die sich Benutzer direkt über LAN oder WAN mit einem ArcGIS Server verbinden können. Diese können sich auch indirekt über eine URL eines Webservice Kataloges mit dem GIS Server verbinden, um die Serverobjekte anzusprechen.
Zusätzlich hat der GIS Server Administrator noch die Möglichkeit, mit Hilfe des ArcCatalog, die Anzahl und Eigenschaften der Serverobjekte zu verändern. Er kann ebenso Serverobjekte hinzufügen, löschen und Output Verzeichnisse festlegen.

Der ArcGIS Server 9.0 besitzt unter anderem zwei wichtige Serverobjekte:[8]

- *MapServer*
- *Geocodeserver*

Die folgende Abbildung soll den Aufbau eines Serverobjektes am Beispiel des *MapServer* verdeutlichen:

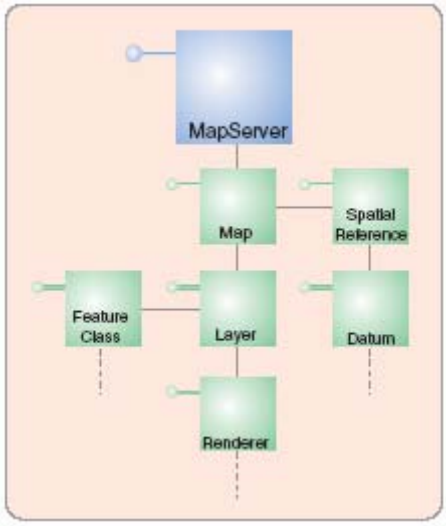

Abb. 9: MapServer Objekt Modell

Quelle: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 40

Arbeitet ein Entwickler mit diesem *MapServer* Objekt, hat er unter anderem auch Zugriff auf das *Map* bzw. *Layer* Objekt einer Karte, mit der auch ein ArcGIS Desktop Entwickler arbeitet. Wenn ein Server Objekt mit Hilfe des ArcCatalog konfiguriert wird, sollten folgende Konfigurationseigenschaften beachtet werden: [9]

- Name des Server Objektes
- Typ des Server Objektes
- Die Initialisierung der Daten und Parameter des Server Objektes
- Pooling oder Unpooling eines Server Objektes
- Wie lange ein Client warten, und das Server Objekt benutzen kann
- Ob das Server Objekt aufbereitet (recycled) wird.

[8] Vgl.: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 39
[9] Vgl. http://www.esri-germany.de/products/arcgis/index.html

3.3.5 Die GIS Server Sicherheit

3.3.5.1 Überblick

Der ArcGIS Server wird als sicherer Server bezeichnet, da er nur die Benutzer zulässt, die von dem ArcGIS Server Administrator authorisiert werden. Es gibt zwei Möglichkeiten, wie man die Sicherheit des ArcGIS Servers gewährleisten kann: [10]

- Innerhalb des GIS Servers
- Innerhalb einer *Web Application* oder eines *Web Service*

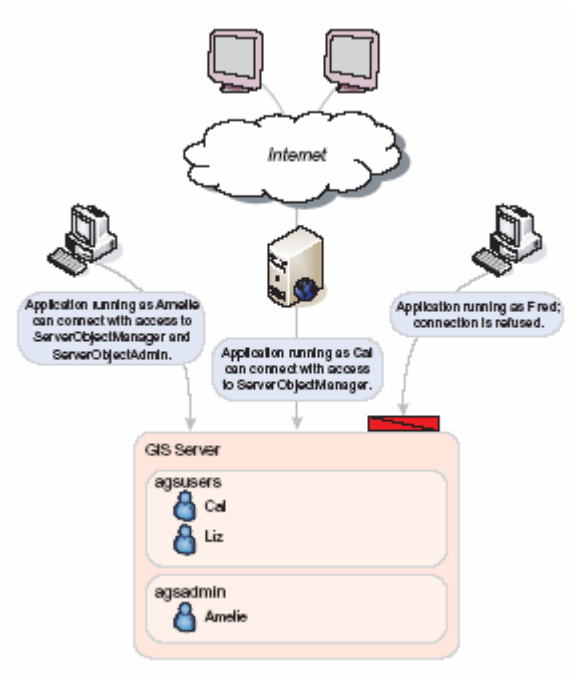

Abb. 10: Sicherheitskomponenten des ArcGIS Server

Quelle: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 46

3.3.5.1.1 Sicherheit innerhalb des GIS Server

Verbindung zum Server haben die Benutzer, die Mitglieder der ArcGIS Server Benutzergruppe (*agsusers*) sind. Diese Benutzer haben dann die Möglichkeit, Server Objekte zu bedienen. Um administrative Rechte zu erhalten, müssen Benutzer als Mitglied der ArcGIS Server Administratorengruppe (*agsadmin*) angemeldet sein.

[10] Vgl.: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 46

Diese administrative Rechte beinhalten:

- Hinzufügen und entfernen von Container Maschinen (*SOC*)
- Hinzufügen, entfernen und modifizieren von Serververzeichnissen
- Hinzufügen, entfernen und modifizieren von Server Objekten
- Starten, stoppen und unterbrechen von Server Objekten
- Betrachten von statistischen Informationen

3.3.5.1.2 Sicherheit innerhalb einer *Web Application* oder *Web Service* (*Impersonation*)

Web Applications und *Web Services* müssen eine *Impersonation* (Personifizierung) benutzen, um sich mit dem GIS Server zu verbinden. Diese *Impersonation* ermöglicht es, ein eigenes Sicherheitsmodell, basierend auf ASP.NET und JAVA, zu entwickeln.
Gegründet auf dieser Sicherheitsstruktur, können anonyme *Web Applications* oder *Web Services* für Benutzer entwickelt werden, die Zugang zu der URL haben.
Dies ermöglicht, eine sichere Anwendung zu entwickeln, die ihre eigenen Benutzer, und unabhängig vom GIS Server, eine Authentisierung und Ermächtigung hat.
Ziel ist es, nicht authorisierten Benutzer, den Zugang zum GIS Server über das Internet zu verweigern. [11]

3.3.6 Der Vergleich zu ArcIMS

Mit dem ArcIMS Server der Firma ESRI werden Geodaten in Form von digitalen Karten oder interaktiven Anwendungen im Internet dargestellt. Diese Daten werden mit einem einfachen HTML-Viewer dargestellt.
Im Vergleich zum ArcGIS Server lässt sich der ArcIMs Server nur bedingt an die Bedürfnisse der Anwender anpassen. Er besitzt keine flexible Programmierungsumgebung wie der ArcGIS Server.
Der große Vorteil des ArcIMS Servers ist jedoch der relativ niedrige Anschaffungspreis im Gegensatz zum ArcGIS Server und die daraus resultierende größre Akzeptanz bei Anwendern und Nutzern.

[11] Vgl.: ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 47

3.4 Arbeitsumgebung

3.4.1 Hardware

Betriebssystem:	Microsoft Windows 2000 Professional
Betriebssystemhersteller:	Microsoft Corporation
Systemname:	R53P99
Prozessor:	2017MHz (~2GHz)
Gesamter realer Speicher:	1,048GB
Gesamter virtueller Speicher:	3,570GB

4 Beschreibung der wichtigsten Objekt Modell Diagramme

4.1 Beschreibung

Objekt-Modellierung ist der Schritt, Objekte als Bausteine der Softwareentwicklung zu verwenden. In diesem Zusammenhang ist mit Objekt eine Klasse gemeint, d.h. die Beschreibung von Gemeinsamkeiten und Abhängigkeiten verschiedener Objekte. Jedes Objekt bzw. Klasse hat seine Identität und Verhalten. Diese Objekte werden verwendet, um Modelle zu entwickeln, in denen die Eigenschaften aller Objekte interagieren und gemeinschaftlich das Gesamtsystem bilden. [12]

Die folgende Abbildung zeigt das ESRI Objekt Modell:

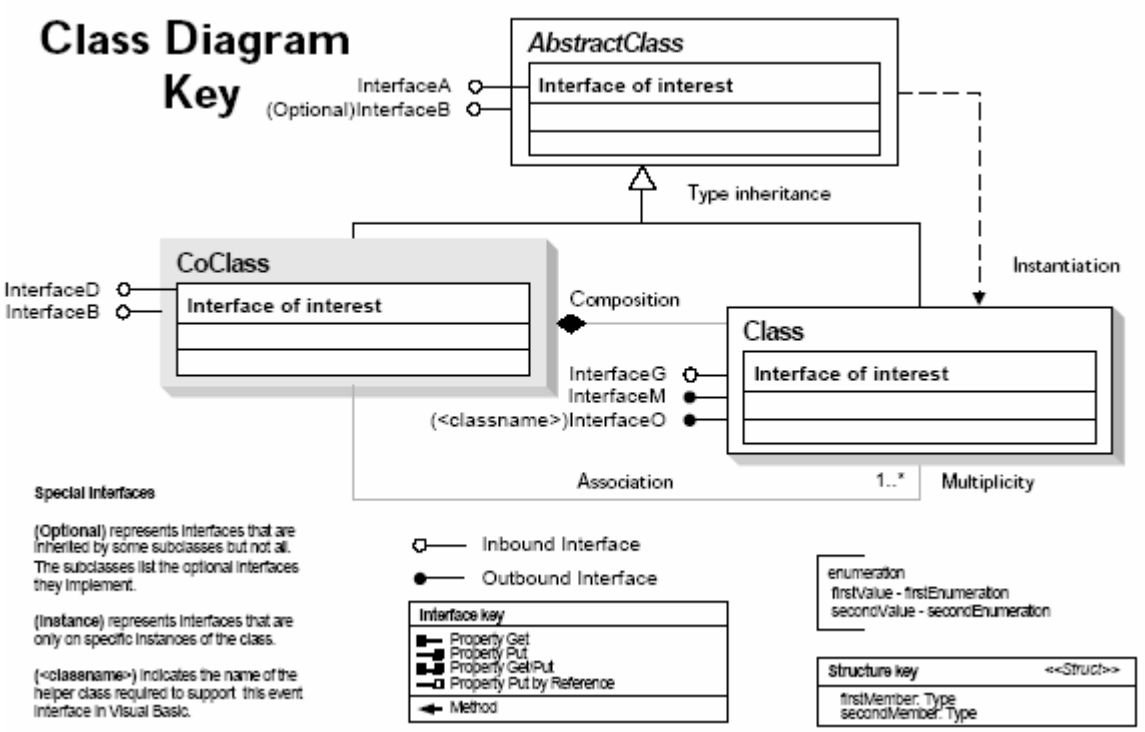

Abb. 11: ESRI Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/ArcGISServer/WebControlsDotNetObjectModel.pdf, Seite 1

Es werden drei Typen von Klassen definiert:

- ***AbstractClass***
  Diese Klasse kann nicht dazu benutzt werden, um neue Objekte zu bilden.
  Sie kann aber durch Vererbung Instanzen von Unterklassen spezifizieren.
- ***CoClass***
  Diese Klasse kann direkt neue Objekte bilden.
- ***Class***
  Diese Klasse kann nicht direkt neue Objekte bilden. Aber Objekte dieser Klasse können als Eigenschaft einer anderen Klasse gebildet werden.

4.2 Das Display Object Model Diagram

4.2.1 Überblick

Die *Display* Bibliothek enthält Objekte, die für die aktuelle Anzeige der GIS Daten verantwortlich sind und beinhaltet auch die Eigenschaften von Symbolen und Farben.
Die Objekte werden in verschiedene Bereiche gegliedert: [i]

- ***Display***
  Stellt die gezeichnete Oberfläche dar.
- ***Colors***
  Die Farben der ArcGIS Anwendungen.
- ***Color Ramps***
  Dient zur Definition der einzelnen Farben.
- ***Symbols***
  Definiert die Darstellung von vier Symbolklassen (Markierungen, Linien, Füllungen, Text).
- ***Display Feedbacks***
  Kontrolliert das Entfernen, Bewegen oder Hinzufügen von graphischen Elementen mit Hilfe der Maus.
- ***Rubber Bands***
  Erzeugen von neuer Geometrie, oder die bestehende Geometrie zu aktualisieren.
- ***Trackers***
  Stellt die Punkte einer Geometrie dar.

---

[12] Vgl.: http://www.tks.at/technologien/objectmodel.html

Bezugnehmend auf die Thematik der Diplomarbeit, wird hier auf das *Display* Objekt näher eingegangen. Die folgende Abbildung zeigt die Architektur des Display Objektes.

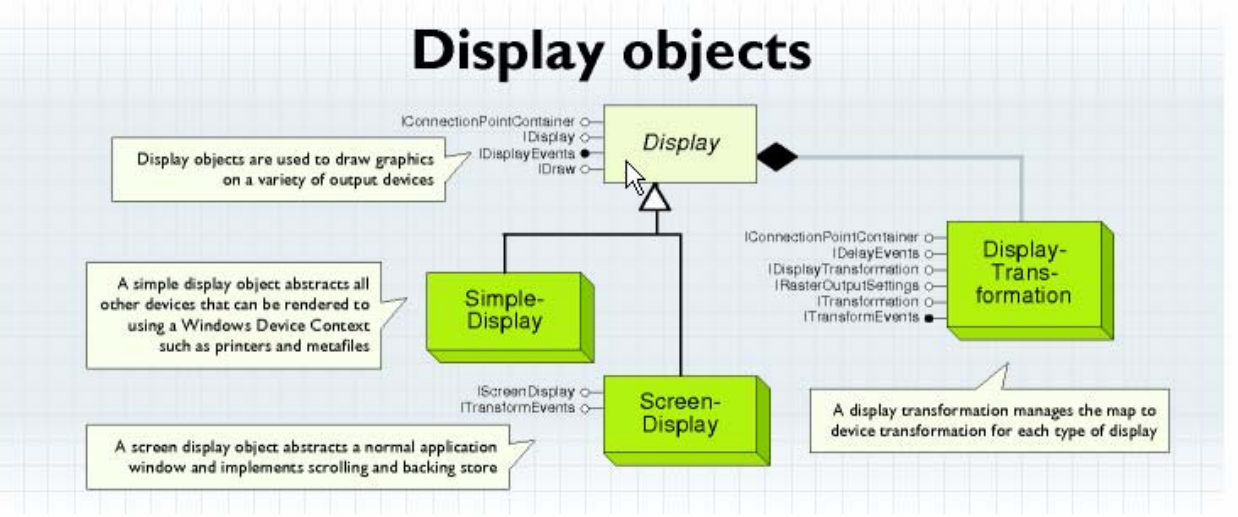

Abb. 12: Das Display Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/esriDisplay/html/Display_overview.htm, DATUM

Das *Display* Objekt abstrahiert eine zeichnende Oberfläche. Jedes Display verwaltet ihr eigenes Transformationsobjekt, welches die Umwandlung von „reale Welt Koordinaten" in den Darstellungsraum, und zurück, beinhaltet. Diese Objekte erlauben dem Anwender, Bilder (*shapes*) auf dem Bildschirm, im Drucker und in exportierten Dateien wiederzugeben. Diese Wiedergabe setzt ein DC (*Device Context*) voraus, der als Schaltzentrale für alle Ausgaben fungiert.[13]

Der DC ist eine Struktur die die Informationen enthält, wie eine Grafik- und Textausgabe zu erfolgen hat. So ist z.B. im DC die aktuelle Zeichenfarbe, oder der gerade aktive Zeichensatz abgelegt. Bevor überhaupt Ausgaben in einem Fenster vorgenommen werden können, muss sich die Anwendung sich immer einen solchen DC holen. Alle Ausgabefunktionen benötigen diesen DC, da er wie gesagt die aktuellen Einstellungen zum Zeichnen enthält. Eine Methode zum Holen eines DC ist zum Beispiel die StartExporting() Methode. Sie liefert als Ergebnis ein Handle auf den DC (HDC) zurück.[14]

[13] Vgl.: ms-help://ESRI.ArcGIS/esriDisplay/html/Display_overview.htm
[14] Vgl.: http://www.cpp-tutor.de/mfc/mfc/kap5/lektion1.htm

Das Display Objekt stellt folgende Standardanzeigen zur Verfügung:

- ***ScreenDisplay***
  Beinhaltet ein normales Anwendungsfenster mit bestimmten Eigenschaften.
- ***SimpleDisplay***
  Beinhaltet alle anderen Geräte, wie zum Beispiel einen Drucker.

Das *DisplayTransformation* Objekt definiert, wie „reale Welt Koordinaten“ in einem Ausgabemechanismus dargestellt werden.
Jedes dieser Objekte beinhaltet Interfaces (Schnittstellen), die wiederum Objekte, Klassen, Methoden und Querverweise zu anderen Objekten und deren Schnittstellen beinhalten. Die relevanten Schnittstellen werden in den nächsten Punkten näher beschrieben.

4.2.2 Beschreibung der relevanten Schnittstellen

4.2.2.1 Das IDisplay Interface

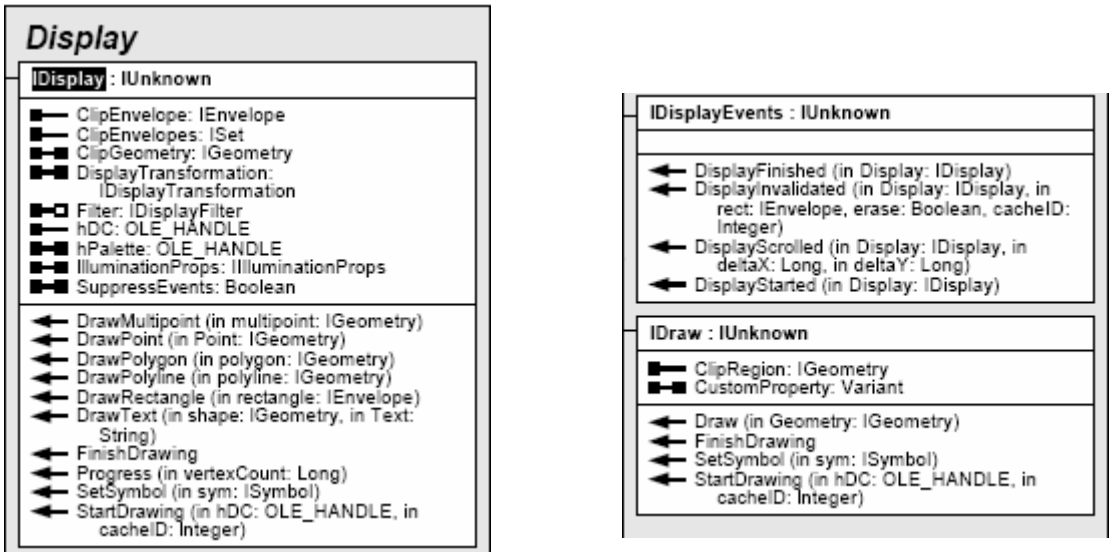

Abb. 13: IDisplay Objekt Modell ESRI

Quelle: ms-help://ESRI.ArcGIS/esriDisplay/DisplayObjectModel.pdf, Seite 1

Das *IDisplay* Interface beinhaltet Methoden und Eigenschaften, die für das Aufzeigen von Punkten, Linien, Polygonen und Text auf einem Ausgabegerät verantwortlich sind. Weiterhin besitzt es Klassen, die auf Querverweise zu anderen Schnittstellen hinweisen.

Die folgende Auflistung soll jeweils nur eine Methode, Eigenschaft und Klasse des Interfaces aufzeigen: [15]

- Die ***DrawPoint()*** Methode
  Diese Methode zeichnet einen bestimmten Punkt auf das Anzeigegerät.
- Die ***ClipEnvelope*** Eigenschaft
  Diese Eigenschaft beinhaltet den Wert der Grenzen einer Region.
- Die ***ScreenDisplay*** Klasse
  Diese Klasse gibt eine Referenz auf das IScreenDisplay Interface weiter.

Der folgende Codeausschnitt soll verdeutlichen, dass die ScreenDisplay Klasse verwendet wird, um eine Referenz auf das *IScreenDisplay* Interface herzustellen:

```
pDT = pActiveView.ScreenDisplay.DisplayTransformation;
```

[15] Vgl.: ms-help://ESRI.ArcGIS/esriDisplay/html/IDisplay.htm

4.2.2.2 Das IScreenDisplay Interface

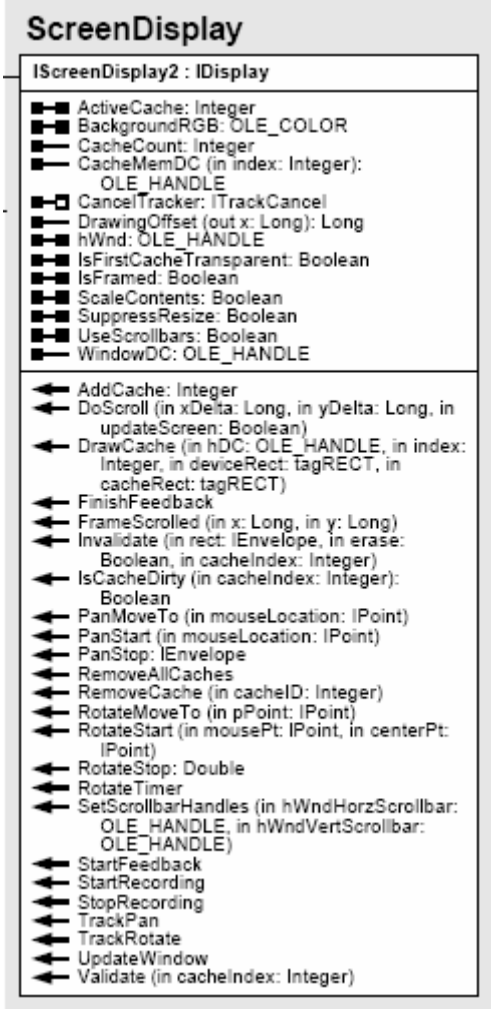

Abb. 14: IScreen Objekt Modell ESRI

Quelle: ms-help://ESRI.ArcGIS/esriDisplay/DisplayObjectModel.pdf, Seite 1

Das *IScreenDisplay* Interface regelt die Attribute der Anzeige eines Bildschirmes.
Dieses Interface besitzt auch eine Anzahl von Methoden, Eigenschaften, Klassen und Querverweise zu anderen Interfaces.

Folgende Auflistung stellt nur die relevanten Methoden und Eigenschaften dar: [16]

- Die ***StartDrawing()*** Methode
  Diese Methode dient zur Vorbereitung des Zeichnens auf das Display.
- Die ***IDisplayTransformation*** Eigenschaft
  Diese Eigenschaft regelt die Konvertierung der Koordinaten zwischen Kartenmaßeinheiten und Ausgabemaßeinheiten. Dient häufig zum Bestimmen der Ausgabekoordinaten.

Der folgende Codeausschnitt soll verdeutlichen, dass die DisplayTransformation Eigenschaft verwendet wird, um eine Referenz auf das *IDisplayTransformation* Interface herzustellen:

```
IDisplaytransformation pDT = pActiveView.ScreenDisplay.DisplayTransformation;
```

4.2.2.3 Das IDisplayTransformation Interface

Wie oben beschrieben, besitzt das *IDisplayTransformation* Interface Methoden und Eigenschaften, um Koordinaten zu konvertieren. Da das Interface eine große Anzahl von Methoden und Eigenschaften aufweist, soll die folgende Auflistung jeweils nur eine Methode, und Eigenschaft des Interfaces aufzeigen:

- Die ***FromMapPoint()*** Methode
  Konvertiert Punkte von Kartenmaßeinheiten zu Ausgabekoordinaten.
- Die ***VisibleBounds*** Eigenschaft
  Diese Eigenschaft steuert den sichtbaren Umfang einer Anzeige. Durch das Einstellen des sichtbaren Umfanges, erhält man Zoom in / Zoom out Effekte.

Der folgende Codeausschnitt soll verdeutlichen, dass die VisibleBounds Eigenschaft verwendet wird, um eine Referenz auf das *IEnvelope* Interface herzustellen, das später anhand des *Geometry* Objektes näher behandelt wird:

```
IEnvelope pEnv = pDT.VisibleBounds as IEnvelope;
```

[16] Vgl.: ms-help://ESRI.ArcGIS/esriDisplay/html/IScreenDisplay.htm

4.2.2.3.1 Die *tagRECT* Struktur der *DeviceFrame* Eigenschaft[17]

Die *DeviceFrame* Eigenschaft spezifiziert den vollen Umfang einer Einheit, deren Ursprung gleich 0 ist. Der Output kann auch auf ein Viereck verwiesen werden, das in einer Einheit liegt, indem man das Viereck als Rahmen deklariert.
In der Programmierpraxis wird meist auf ein Kartenobjekt (IMXDocument) verwiesen, das den vollen Umfang eines Kartenfensters zurückgibt. Unterschiede gibt es zwischen der Kartenansicht (*Map view*) und der Layoutansicht (*Layout view*) in einem MXDokument. Mit der *DeviceFrame* Eigenschaft bekommt der User die aktuelle Größe der Karte. In der Layoutansicht bekommt man die Größe des Rahmens, in dem die Karte liegt.

Mit der *tagRECT* Struktur kann man die Werte der vier Koordinatenpunkte speichern. Der folgende Codeausschnitt soll dies verdeutlichen:

```
tagRECT exportRECT;

exportRECT.left = 0;
exportRECT.top = 0;
exportRECT.right = pActiveView.ExportFrame.right * (imageRes / iScreenResolution);
exportRECT.bottom = pActiveView.ExportFrame.bottom * (imageRes / iScreenResolution);
```

Die Variable *exportRECT* wird als *tagRECT* deklariert. Der Ursprung ist gleich 0. Auf die Koordinate rechts unten wird ein Viereck aufgezogen. Die Referenzierung erfolgt über das *Carto* Objekt. Dieses wird in einem späteren Abschnitt behandelt.

4.3 Das Geometry Object Model Diagram

4.3.1 Überblick

Die *Geometry* Bibliothek beinhaltet die Eigenschaften von Geometrie, Formen und anderen graphischen Elementen. Die grundlegenden Eigenschaften, auf die der Benutzer einwirken kann, sind *Points*, *Multipoints*, *Polylines* und *Polygone*.
Polylines und Polygone bestehen aus einer Reihenfolge von verbundenen Segmenten, die einen Pfad bilden. Ein Segment besteht aus zwei unterschiedlichen Punkten, der Anfang und der Endpunkt und eine Elementart, die die Kurve am Anfang definiert.

[17] Vgl.:ms-help://MS.VSCC.2003/ESRI.ArcGIS/esriDisplay/html/IDisplayTransformation_DeviceFrame.htm

Geometrien können durch räumliche Hinweise (spatial references) auf die reale Welt georeferenziert werden. Ein räumlicher Hinweis definiert das Koordinatensystem und das Gebiet der Geometrie. [18]

Entwickler können das räumliche Bezugssystem verlängern, indem sie neue räumliche Hinweise definieren.

4.3.2 Beschreibung der relevanten Schnittstellen

4.3.2.1 Das IEnvelope Interface

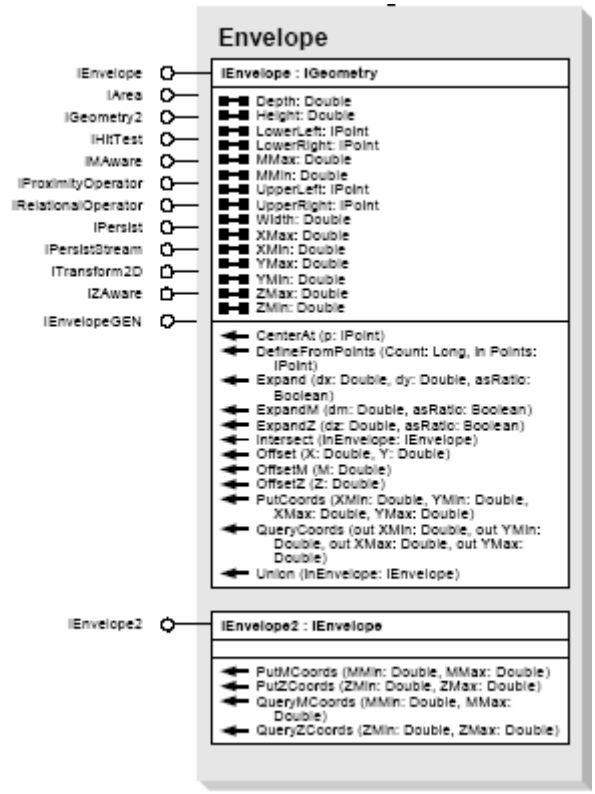

Abb. 15: IEnvelope Objekt Modell ESRI

Quelle: ms-help://ESRI.ArcGIS/esriGeometry/GeometryObjectModel.pdf, Seite 1

Ein Envelope ist ein rechteckiges Fenster, das ein spezifisches Element beinhaltet. Envelopes können als Ansichtsbereich für einen Datenrahmen dienen.

[18] Vgl.: ms-help://ESRI.ArcGIS/esriGeometry/html/Geometry_overview.htm

Folgende Auflistung zeigt nur die relevante Methode und Eigenschaft:[19]

- Die ***Height*** Eigenschaft
  Definiert die Höhe des Envelopes in Pixel.
- Die ***PutCoords()*** Methode
  Konstruiert ein Envelope mit den 4 Eckpunkt Koordinaten.

Die verschiedenen Möglichkeiten eines *Envelope* Objekts werden in der folgenden Abbildung bildlich dargestellt:

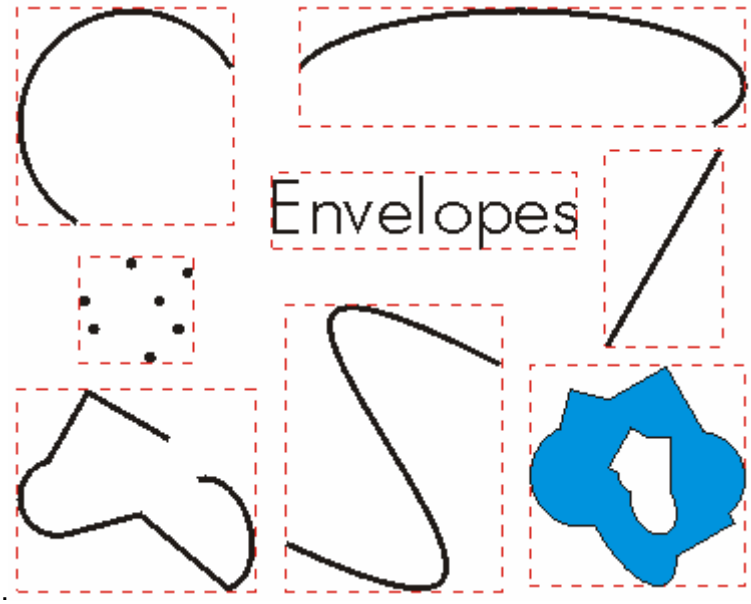

Abb. 16: Envelope Objekt

Quelle: ms-help://ESRI.ArcGIS/esriGeometry/html/IGeometry_Envelope.htm

[19] Vgl.: ms-help://ESRI.ArcGIS/esriGeometry/html/IEnvelope.htm

Die *PutCoords()* Methode mit den 4 Eckpunkt Koordinaten *XMin*, *YMin*, *XMax* und *YMax* in bildlicher Darstellung:

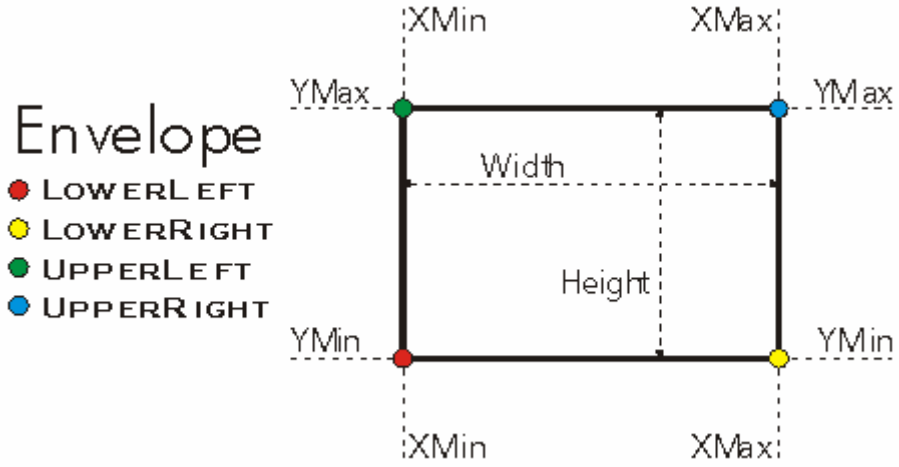

Abb. 17: Die PutCoords() Methode

Quelle: ms-help://ESRI.ArcGIS/esriGeometry/html/IEnvelope_PutCoords.htm

Der folgende Codeausschnitt stellt die *PutCoords()* Methode dar, die die Koordinaten eines Envelopes konstruiert:

```
IEnvelope pPixelBoundsEnv;
pPixelBoun-
sEnv.PutCoords(exportRECT.left,exportRECT.top,exportRECT.right,exportRECT.bottom);
```

pPixelBoundsEnv wird als Variable von IEnvelope gesetzt.

4.4 Carto Object Model Diagram

4.4.1 Überblick[20]

Die Carto Bibliothek beinhaltet das Entwickeln und Anzeigen von Karten, welche auch mit mehreren Karten zusammen auf einer Seite angezeigt werden können.
Die Carto Bibliothek besitzt eine Vielzahl von Objekten, die das Verhalten einer Karte und Seite darstellen. Dies lässt die Aussage zu, dass die Carto Bibliothek eine der wichtigsten Bibliotheken für ArcGIS Programmierer ist.

Nachfolgend werden die Objekte der Bibliothek aufgelistet:

- ***MapElements***
  Beinhaltet zum Beispiel Nordpfeil, Maßstabsleiste, Bilder usw..
- ***Map & PageLayout***
  Beinhaltet Eigenschaften der Karte und des Layouts.
- ***Map Surrounds***
  Sind spezifische Elemente, die mit einem Kartenobjekt verbunden sind.
- ***Map Grids***
  Beinhaltet die Rastereigenschaften eines Kartenobjektes.
- ***Renderers***
  Objekte, die ArcGIS Datenschichten speichern.
- ***Labeling***
  Eigenschaften, die das Platzieren eines Textes beinhalten.
- ***Annotation***
  Zeichnen Text- und Graphikelemente.
- ***Dimension***
  Maße, die spezifische Längen oder Abstände eines Kartenobjektes beinhalten.
- ***Layers***
  Beinhalten verschiedene Schichten von geographischen Informationen der Kartenobjekte.
- ***MapServer***
  Objekt, das ArcGIS Dokumente im Internet bzw. Intranet darstellt.

[20] Vgl.:ms-help://ESRI.ArcGIS/esriCarto/html/Carto_overview.htm

4.4.2 Beschreibung der relevanten Schnittstellen

4.4.2.1 Das IActiveView Interface

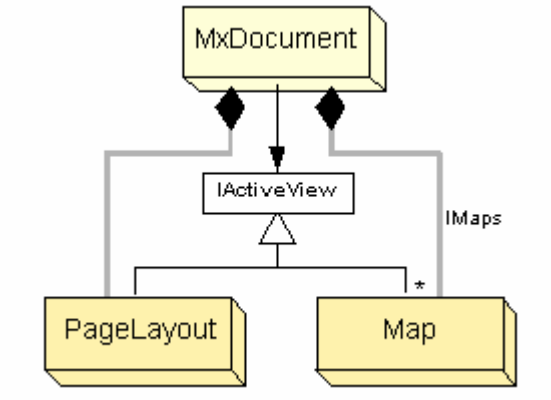

Abb. 18: Die IActiveView Struktur

Quelle: ms-help://ESRI.ArcGIS/esriCarto/html/IActiveView.htm, DATUM: 09.03.2005

Diese Schnittstelle verwaltet das Hauptanwendungsfenster in ArcMap.
Das *IActiveView* Interface beinhaltet zwei Objekte:

- ***Map*** Objekt
- ***PageLayout*** Objekt

Diese zwei Objekte entsprechen den zwei unterschiedlichen Ansichten, der Layout- und Datenansicht, in ArcMap.
Relevant für die Diplomarbeit ist das *PageLayout*, welches in einem späteren Abschnitt näher behandelt wird.

Diese Abbildung zeigt das *IActiveView* Objekt Modell:

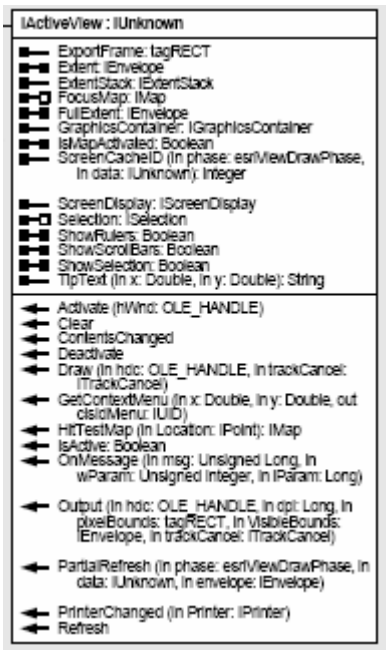

Abb. 19: Die IActiveView Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 1

Die folgende Auflistung beinhaltet nur die relevante Methode und Eigenschaft des Interfaces:

- Die ***ExportFrame*** Eigenschaft
  Gibt den Wert eines zu exportierendes Gebietes zurück.
  Die ***Output()*** Methode
  Übergibt die zu exportierenden Werte in ein Ausgabegrät.

4.4.2.1.1 Die *ExportFrame* Eigenschaft

Diese Eigenschaft übergibt ein zu exportierendes Rechteck, das auf die *tagRECT* Struktur referenziert ist.
Der folgende Codeausschnitt stellt die *ExportFrame* Eigenschaft dar, die die Werte des Rechtecks einer aktiven Ansicht übergibt:

```
tagRECT exportRECT;

exportRECT.left = 0;
exportRECT.top = 0;
exportRECT.right = pActiveView.ExportFrame.right * (imageRes / iScreenResolution);
exportRECT.bottom = pActiveView.ExportFrame.bottom * (imageRes / iScreenResolution);
```

4.4.2.1.2 Die Output() Methode

Wie oben beschrieben, übergibt diese Methode die zu exportierenden Parameter in ein Ausgabegerät. Diese Parameter werden in der folgenden Auflistung einzelnd beschrieben:[21]

- ***hdc***
  Spricht das Ausgabegerät an.
- ***dpi***
  Beinhaltet die Auflösung in dots per inch.
- ***pixelBounds***
  Werte des Rechtecks in Pixel.
- ***VisibleBounds***
  Beinhaltet den Zoomumfang.

Diese Parameterübergabe wird anhand des folgenden Codeauschnittes deutlich:

```
pActiveView.Output(hDC, imageRes, ref exportRECT, pEnv, pCancel);
```

4.4.2.2 Das IPageLayout Interface

Das *IPageLayout* Interface modifiziert das Layout der Seite, welches Eigenschaften der Graphikpositionierung und Methoden besitzt, wie die Seite auf dem Bildschirm angezeigt wird.[22]
Die folgende Abbildung zeigt das Objekt des *PageLayout* Objektes:

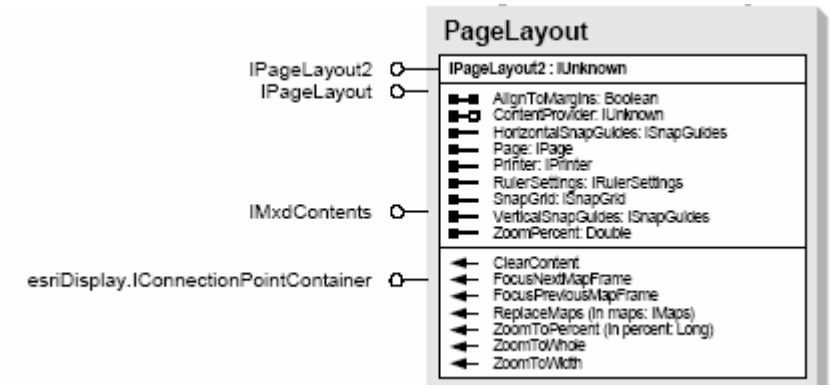

Abb. 20: Das PageLayout Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 1

[21] Vgl.: ms-help://ESRI.ArcGIS/esriCarto/html/IActiveView_Output.htm
[22] Vgl.: ms-help://ESRI.ArcGIS/esriCarto/html/IPageLayout.htm

Da auch hier das Interface eine große Anzahl an Eigenschaften, Methoden und Klassen aufweist, werden nur die relevanten Objekte aufgelistet:

- Die ***Page*** Eigenschaft
  Referenz auf das IPage Interface, welches die Printer Seite repräsentiert.
- Die ***PageLayout*** Klasse
  Gibt einen Verweis auf das *PageLayout* Objekt.

Der folgende Codeausschnitt soll einen Verweis auf das PageLayout Objekt verdeutlichen:

```
IActiveView pActiveView = webPageLayout.PageLayout as IActiveView;
```

Im obigen Beispiel wird von dem *webPageLayout* Objekt auf das *IPageLayout* Interface referenziert.

4.4.2.3 Das IPage Interface

IPage ist die Primärschnittstelle des Pageobjektes, welches die Seiteneigenschaften des Layouts im ArcMap, zum Beispiel die Orientation und die Seitengrösse bestimmt.
Die folgende Abbildung zeigt das Objekt Modell eines *IPage* Interfaces:

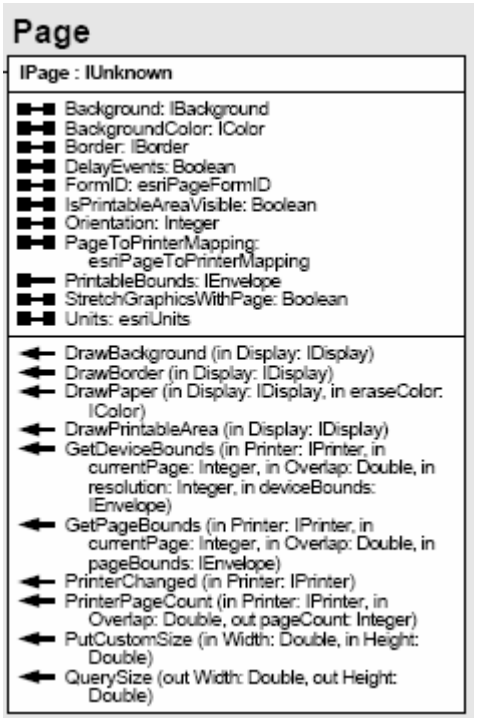

Abb. 21: Das IPage Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 1

Die folgende Auflistung soll nur die relevanten Eigenschaften und Methoden des *IPage* Interfaces aufzeigen:[23]

- Die ***FormID*** Eigenschaft
  Beinhaltet das Seitenformat.
- Die ***Orientation*** Eigenschaft
  Beinhaltet, ob die Seite im Hoch- oder Querformat gesetzt wird.
- Die ***Units*** Eigenschaft
  Bestimmt die speziellen Maßeinheiten der Seite.
- Die ***QuerySize()*** Methode
  Bestimmt die Größe der Seite in bestimmte Maßeinheiten.

[23] Vgl.: ms-help://ESRI.ArcGIS/esriCarto/html/IPage.htm

4.4.2.3.1 Die *FormID* Eigenschaft

Die *FormID* Eigenschaft referenziert auf die *EsriPageFormID* Konstante.
Diese Konstante beinhaltet die verschiedenen Seitenformate. Die folgende Tabelle zeigt nur die relevanten Formate:

Konstante	Wert	Beschreibung
esriPageFormA4	7	Metric A4 - 210mm x 297mm.
esriPageFormA3	8	Metric A3 - 297mm x 420mm.
esriPageFormA2	9	Metric A2 - 420mm x 594mm.
esriPageFormA1	10	Metric A1 - 594mm x 841mm.
esriPageFormA0	11	Metric A0 - 841mm x 1189mm.

Tabelle 1: EsriPageFormID
Quelle: ms-help://MS.VSCC.2003/ESRI.ArcGIS/esriCarto/html/esriPageFormID.htm,

4.4.2.3.2 Die *Orientation* Eigenschaft

Die *Orientation* Eigenschaft beinhaltet, ob die Seite im Hoch- oder Querformat dargestellt werden soll. Die folgenden Variablen werden vergeben:

1 = ***portrait***
2 = ***landscape***

4.4.2.3.3 Die *Units* Eigenschaft

Die *Units* Eigenschaft wird auf *esriUnits* Konstante referenziert.
Folgende tabellarische Auflistung zeigt die einzelnen Werte:

Konstante	Wert	Beschreibung
esriInches	1	Inches
esriMillimeters	7	Millimeter
esriCentimeters	8	Centimeter
esriMeters	9	Meter

Tabelle 2: esriUnits
Quelle: ms-help://MS.VSCC.2003/ESRI.ArcGIS/esriSystem/html/esriUnits.htm

4.4.2.3.4 Die *QuerySize()* Methode

Die *QuerySize()* Methode bestimmt die Seitengröße in bestimmte Einheiten.
Folgende Variablen werden dafür vergeben:

- ***Width***
- ***Height***

4.5 Die Output Bibliothek

4.5.1 Überblick

Die *Output* Bibliothek kann man in Bezug zur Diplomarbeit als wichtige Bibliothek bezeichnen, da sie den graphischen Output an bestimmte Geräte, wie z.B. Printer und Plotter, und in bestimmte Dateiformate, wie z.B. PDF und JPEG, beinhaltet.
Die *Output* Bibliothek gliedert sich in zwei Bereiche:[24]

- ***Printers***
  Die Printerobjekte stellen die Verbindung zu den einzelnen Printmaschinen her.
  Jede Maschine verarbeitet zeichnende Anweisungen zu einem Format, das auf einen Drucker übertragen werden kann. Jedes dieser Objekte kontrolliert z.B. die PostScript Schrift und die Papiergröße.
- ***Exports***
  Diese Exportobjekte liefern Zugang zu den graphischen Exporttreibern, welche Eigenschafte und Methoden bzgl. Auflösung, Kompression usw. kontrollieren.

Die *Image-Export* Objekte übertragen zeichnende Befehle in eine Bitmap, die in eines von den fünf folgenden unterschiedlichen Rasterbildformaten weitergeführt werden kann:

- ***BMP***
  Windows Bitmap Format.
- ***JPEG***
  Joint Photographic Experts Group Format.
- ***PNG***
  Portable Network Graphics Format.

- ***TIFF***

  Tagged Image File Format.
- ***GIF***

  Graphics Interchange Format.

Die ***Vektor-Export*** Objekte übersetzen zeichnende Befehle in eines der fünf folgenden Vektorformate:[25]

- ***EMF***

  Windows Enhanced Metafile Format.
- ***EPS***

  Encapsulated PostScript Format.
- ***AI***

  Adobe Illustrator Format.
- ***PDF***

  Portable Document Format .
- ***SVG***

  Scalable Vector Graphics Format.

[24] Vgl.: ms-help://ESRI.ArcGIS/esriOutput/html/IExport.htm
[25] Vgl.: ms-help://ESRI.ArcGIS/esriOutput/html/Output_library.htm

4.5.2 Beschreibung der relevanten Schnittstellen

4.5.2.1 Das IExport Interface

Die folgende Abbildung beinhaltet das *IExport* Objekt Modell:

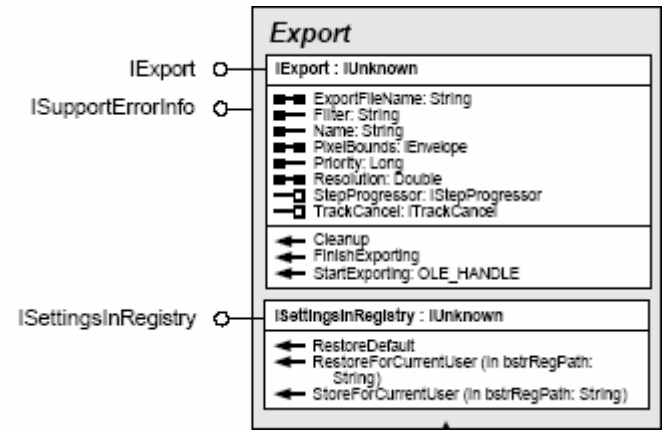

Abb. 22: Das IExport Objekt Modell

Quelle: ms-help://ESRI.ArcGIS/esriOutput/OutputObjectModel.pdf, Seite 1

Die folgende Auflistung stellt nur die relevanten Eigenschaften, Methoden und Klassen des *IExport* Interfaces dar:[26]

- Die ***ExportFileName*** Eigenschaft
  Beinhaltet das Zielverzeichnis und den Namen der zu exportierenden Datei.
- Die ***PixelBounds*** Eigenschaft
  Beinhaltet die Grenzen in Pixel, der Exportoberfläche.
- Die ***Resolution*** Eigenschaft
  Referenziert die Auflösung, in DPI.
- Die ***StartExporting()*** Methode
  Initialisiert den Exporter zum Starten.
- Die ***FinishExporting()*** Methode
  Initialisiert den Exporter zum Beenden.
- Die ***TrackCancel()*** Methode
  Dies ermöglicht den Benutzer, mit Hilfe einer Eingabetaste das Exportieren zu stoppen.

[26] Vgl.: ms-help://ESRI.ArcGIS/esriOutput/html/IExport.htm

- Die ***ExportPDF*** Unterklasse
  Diese Unterklasse erstellt ein neues Export Objekt.

Auf das Scripting der Methoden und Eigenschaften dieses Interfaces wird im Kapitel 6 (Abschnitt 6.2.1.2.1) näher eingegangen.

4.6 Die ESRI.ArcGIS.Server Bibliothek

4.6.1 Überblick

Die Server Bibliothek beinhaltet Objekte, die den Zugriff, und die Arbeit mit dem Server erlauben. Die Entwickler erhalten über das *GISServerConnection* Objekt Zugriff auf den GIS Server. Dieses Objekt erlaubt wiederum Zugriff auf den *ServerObjectManager.*
Über diese Objekte können nun Entwickler, mit Hilfe des *ServerContext* Objektes, Methoden, Eigenschaften und Klassen bearbeiten.[27]
Wichtig für das Projekt ist das *ServerContext* Objekt, das im folgenden Abschnitt näher behandelt wird.

4.6.2 Das IServerContext Interface

Der *ServerContext* ist ein reservierter Raum innerhalb des Servers, der eine Anzahl von laufenden Objekten beinhaltet. Diese sind ArcObjects, die vom Entwickler erzeugt werden.[28]
Im folgenden Code-Ausschnitt wird ein *ServerContext* mit Hilfe eines vordefinierten Session Objektes in eine Variable referenziert:

```
m_ctx = sessobj as IServerContext;

webPageLayout = new WebPageLayout(m_ctx, m_host, pageDescription);
```

Der *ServerContext* wird als Parameter für das Erzeugen einer neuen Instanz der *WebPageLayout* Klasse benötigt.

[27] Vgl.: ms-help://ESRI.ArcGIS/esriServer/html/Server_overview.htm
[28] Vgl.: ms-help://ESRI.ArcGIS/esriServer/html/IServerContext.htm, DATUM

4.7 Der ESRI.ArcGIS.Server.WebControls Namespace

Der ESRI.ArcGIS.Server.WebControls Namespace enthält .NET ADF Web Controls (Web Elemente). Der Namespace (Namensraum) enthält auch Klassen, die das Hinzufügen von erweiterten Funktionalitäten bzgl. der WebControls erlauben.[29]
Da der Namespace eine große Anzahl von Klassen aufweist, werden nur die folgenden, relevanten Klassen näher betrachtet:

- Die ***ServerConnection*** Klasse
  Definiert die Verbindung zu einem Server Object Manager des ArcGIS Servers.
- Die ***WebPageLayout*** Klasse
  Erleichtert das Arbeiten mit einer Layoutansicht eines Map Dokumentes.

4.7.1 Die *WebPageLayout* Klasse

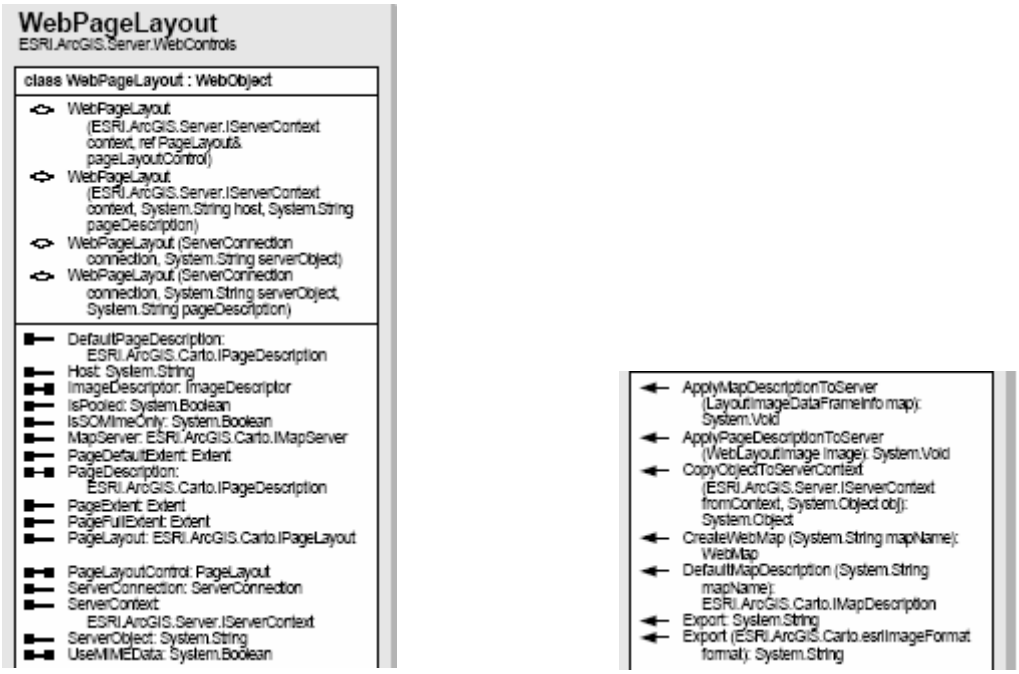

Abb. 23: Das Objekt Modell der WebPageLayout Klasse

Quelle: ms-help://ESRI.ArcGIS/ArcGISServer/WebControlsDotNetObjectModel.pdf, Seite 1

[29] Vgl.: ms-help://ESRI.ArcGIS/ESRI.ArcGIS.Server.WebControls.html, DATUM

Die WebPageLayout Klasse beinhaltet zudem die Navigation des Layouts und des Datenrahmens.
Die folgende Auflistung bezieht sich nur auf die relevante Methode, Eigenschaft und Konstruktor:[30]

- Die ***PageLayout*** Eigenschaft
  Gibt eine Referenz auf das *IPageLayout* Interface der Carto Bibliothek.
- Die ***Export()*** Methode
  Exportiert das Layout in seiner vollen Größe in ein spezifiziertes Image Format.
- Der ***WebPageLayout()*** Konstruktor
  Initialisiert eine neue Instanz der *WebPageLayout* Klasse.

[30]Vgl.: ms-help://ESRI.ArcGIS/ESRI.ArcGIS.Server.WebControls/ WebPageLayoutMembers.html

4.7.1.1 Die *Export()* Methoden

4.7.1.1.1 Der Export mit der *ESRI.ArcGIS.Carto.esriImageFormat* Enumeration

Um ein qualitativ hohes Printformat zu generieren ist es unabdingbar, die verschiedenen Export Möglichkeiten der ArcGIS Bibliotheken zu entwickeln und zu testen.
Die Export Methode mit der Bibliothek der *ESRI.ArcGIS.Carto.esriImageFormat* Enumeration (Aufzählung) der *WebPageLayout* Klasse, bietet den Export in eine Vielzahl von Formaten an. Die folgende Code-Abbildung zeigt eine definierte Format Methode, welche eine Auswahl von verschiedenen Formaten zulässt:

```
private ESRI.ArcGIS.Carto.esriImageFormat convertToImageFormat(string imgFormat)
{
    ESRI.ArcGIS.Carto.esriImageFormat newFormat = EXPORT_IMAGE_FORMAT;
    switch (imgFormat)
    {
        case "GIF":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImageGIF;
            break;
        case "JPG":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImageJPG;
            break;
        case "PDF":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImagePDF;
            break;
        case "PS":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImagePS;
            break;
        case "SVG":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImageSVG;
            break;
        case "TIFF":
            newFormat = ESRI.ArcGIS.Carto.esriImageFormat.esriImageTIFF;
            break;
    }

    return newFormat;
}
```

Abb. 24: Code-Ausschnitt der Format Methode

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 252 - 278

Der folgende Code-Ausschnitt zeigt die Implementation der Format Methode in die Export Methode der WebPageLayout Klasse, die vorher neu erzeugt wurde:

```
string url = webPageLayout.Export(EXPORT_IMAGE_FORMAT);
```

Diese Möglichkeit lässt allerdings keine Veränderung der Auflösung zu.
Zusammenfassend bietet es dem Benutzer keine flexible Exportmöglichkeit, das für das Ziel der Diplomarbeit als nicht ausreichend bewertet werden kann.

4.7.1.2 Der *WebPageLayout()* Konstruktor

Wie schon beschrieben, wird dadurch eine neue *WebPageLayout* Instanz erzeugt.
Diese Instanz beinhaltet den ServerContext, den Host und die abgelegte Session.
Auf diese Punkte wird später näher eingegangen.
Folgender Code-Ausschnitt zeigt die Erzeugung einer neuen *WebPageLayout* Instanz:

```
webPageLayout = new WebPageLayout(m_ctx, m_host, pageDescription);
```

4.8 Der System.Web.SessionState Namespace

4.8.1 Beschreibung

Der *System.Web.SessionState* Namespace stellt Klassen und Schnittstellen bereit, die die Speicherung von Daten ermöglichen, welche für einen einzelnen Client innerhalb einer Webanwendung auf dem Server spezifisch sind. Die Sitzungsstatusdaten werden verwendet, um für den Client eine permanente Verbindung mit der Anwendung darzustellen. Die Zustandsinformationen können innerhalb des lokalen Prozessspeichers gespeichert werden oder bei Webfarmkonfigurationen prozessextern mit Hilfe des ASP.NET-Statusdienstes oder einer SQL Server-Datenbank.
Der Sitzungsstatus kann bei Clients verwendet werden, die keine Cookies unterstützen. ASP.NET kann so konfiguriert werden, dass eine Sitzungs-ID in der zwischen Client und Server übermittelten URL-Zeichenfolge verschlüsselt wird.[31]

[31] Vgl.: http://msdn.microsoft.com/library/deu/cpref/html/frlrfsystemwebsessionstate.asp

4.8.1.1 Die *HttpSessionState* Klasse

Die *HttpSessionState* Klasse ermöglicht den Zugriff auf Werte des Sitzungsstatus und Einstellungen der Sitzungsebene sowie auf Methoden für die Lebensdauerverwaltung.

Der folgende Code Ausschnitt beinhaltet das Abrufen einer laufenden Session (Sitzung):

```
//Abrufen und Speicherung der Session
Session["MailToAddress"] = MailToAddress.Text;
```

4.9 Der System.Web.Mail Namespace

4.9.1 Beschreibung

Der *System.Web.Mail* Namespace enthält Klassen, die das Erstellen und Senden von Nachrichten mit Hilfe der CDOSYS-Nachrichtenkomponente (Collaboration Data Objects for Windows 2000) ermöglichen. Die E-Mail-Nachricht wird durch den in Microsoft Windows 2000 integrierten SMTP-Maildienst oder einen beliebigen SMTP-Server übermittelt. Die Klassen des Namespaces können von ASP.NET oder einer beliebigen verwalteten Anwendung verwendet werden.[32]
Die folgende Auflistung zeigt die relevanten Klassen, die der Namespace enthält:

- Die ***MailMessage*** Klasse
  Stellt Eigenschaften und Methoden zum Erstellen einer E-Mail-Nachricht bereit.
- Die ***SmtpMail*** Klasse
  Stellt Eigenschaften und Methoden für das Senden von Nachrichten mit Hilfe der *CDOSYS*-Nachrichtenkomponente bereit.

[32] Vgl.: http://msdn.microsoft.com//library/DEU/cpref/html/frlrfsystemwebmail.asp

4.9.1.1 Die *MailMessage* Klasse

Folgende Eigenschaften sind für das Versenden der Print Datei erforderlich:

- Die ***From*** Eigenschaft
  Ruft die E-Mail-Adresse des Absenders ab oder legt diese fest.
- Die ***To*** Eigenschaft
  Ruft eine durch Semikolon getrennte Liste mit E-Mail-Adressen von Empfängern ab, oder legt diese fest.
- Die ***Subject*** Eigenschaft
  Ruft die Betreffzeile der E-Mail-Nachricht ab oder legt diese fest.
- Die ***Body*** Eigenschaft
  Ruft den Textkörper der E-Mail-Nachricht ab oder legt diesen fest.
- Die ***BodyFormat*** Eigenschaft
  Ruft den Inhaltstyp des Textkörpers der E-Mail-Nachricht ab oder legt diesen fest.

Folgende Abbildung stellt einen Code Ausschnitt der Methode zum Versenden einer E-Mail dar:

```
private void SendMail(string address, string message)
{

    MailMessage MyMail = new MailMessage();
    MyMail.From = m_MailFromAddress;
    MyMail.To = address;
    MyMail.Subject = "Ihre Datei wurde exportiert";
    MyMail.Body = message;
    MyMail.BodyFormat = MailFormat.Html;
    SmtpMail.SmtpServer = m_MailServer;
    SmtpMail.Send(MyMail);

}
```

Abb. 25: Code-Ausschnitt der Mail Methode

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 239 - 251

4.9.1.2 Die *SmtpMail* Klasse

Die E-Mail-Nachricht kann durch den in Microsoft Windows 2000 integrierten SMTP-Maildienst oder einen beliebigen SMTP-Server übermittelt werden. Die Typen im *System-WebMail*-Namespace können von ASP.NET oder einer beliebigen verwalteten Anwendung verwendet werden.

Wenn die *SmtpServer*-Eigenschaft nicht festgelegt ist, werden die E-Mail-Nachrichten gemäß Standardeinstellung in eine Warteschlange des Windows 2000-Systems gestellt, damit das aufrufende Programm nicht das Netzwerk blockiert. Wenn die SmtpServer-Eigenschaft festgelegt ist, werden die E-Mail-Nachrichten direkt an den angegebenen Server übermittelt.[33]

Folgende relevante Eigenschaft und Methode beinhaltet diese Klasse:

- Die ***SmtpServer*** Eigenschaft
  Ruft den Namen des SMTP-Relay-Mailservers ab, über den E-Mail-Nachrichten gesendet werden sollen, oder legt diesen fest.
- Die ***Send()*** Methode
  Sendet eine E-Mail-Nachricht.

Folgender Codeausschnitt soll dies noch einmal verdeutlichen:

```
SmtpMail.SmtpServer = m_MailServer;
SmtpMail.Send(MyMail);
```

[33] Vgl.: http://msdn.microsoft.com/ library/DEU/cpref/html/frlrfsystemwebmailsmtpmailclasstopic.asp

5 Planung und Entwicklung der einzelnen „*Print on demand*" Komponenten

5.1 Die Planung

Zur der Realisierung eines Projektes gehört zunächst eine gründliche Planung.
Ziel und Zweck einer Planung ist es, die Anforderungen eines Projektes organisatorisch zu gliedern, um anhand von Teilschritten das Ziel zu erreichen. Der Vorteil besteht darin, den Umfang eines Projektes klar abzuschätzen.
Die Planung dieses Projektes gliedert sich in folgende Bereiche:

- Einarbeitung in die Programmiersprachen Visual C#, VB .NET, ASP .NET, XML und in die Entwicklungsumgebung von Visual Studio .NET
- Einarbeitung in die Objekt Modelle und ArcObjects Programmierung
- Test von kleinen Code-Beispielen aus der ArcGIS Developer Help
- Realisierung und Testen der Funktionen.
- Ergebnis

5.1.1 Einarbeitung in die Programmiersprachen Visual C#, VB .NET, ASP .NET und XML.

Für die Entwicklung von Web Applikationen genügt es nicht, sich nur auf eine Programmiersprache zu vertiefen. Durch das Zusammenspiel von Browser und Server müssen mehrere Programmiersprachen berücksichtigt werden.

ASP .NET (Active Server Pages) ist eine Programmiersprache für Webserver von Microsoft innerhalb des .NET Frameworks. ASP-Seiten enthalten neben den üblichen HTML-Tags Programmstrukturen (Skripte), die von einem Server ausgewertet werden. ASP-Skripte dienen zum Erstellen dynamischer Webseiten, die z.B. an eine Datenbank angebunden sind. [34]

VB .NET ist eine Programmiersprache für die Microsoft Windows Umgebung.
Mit dieser Sprache lässt sich sowohl objektorientiert, als auch Webanwendungen mit Hilfe von *ASP .Net* programmieren.

[34] Vgl.: http://www.galileocomputing.de/glossar/gp/

XML (eXtensible Markup Language) ist eine Metasprache für die Definition von Dokumenttypen, die ein anpassungsfähiges Datenformat für den Austausch strukturierter Dokumente im Web ermöglicht. Sie gibt dem Entwickler die Möglichkeit, eigene angepasste Markup-Sprachen - quasi Dialekte - mit eigenen Tags für viele verschiedene Dokumenttypen zu definieren.[35]

Für die Einarbeitung in die Programmiersprachen wurden hauptsächlich Bücher, E-Books und Internet-Foren benutzt.

[35] http://www.galileocomputing.de/glossar/gp/

5.1.1.1 Einarbeitung in die Objekt Modelle und ArcObjects Programmierung.

ArcGIS besitzt eine Developer Help Dokumentation, in der man Anleitungen, Beispiele und Objekt Modelle findet. Außer im ESRI Forum, findet man keine andere Literatur, die speziell für die serverseitige Programmierung des ArcGIS Servers veröffentlicht wurde. Folgende Abbildung zeigt die ArcGIS Developer Help von ESRI:

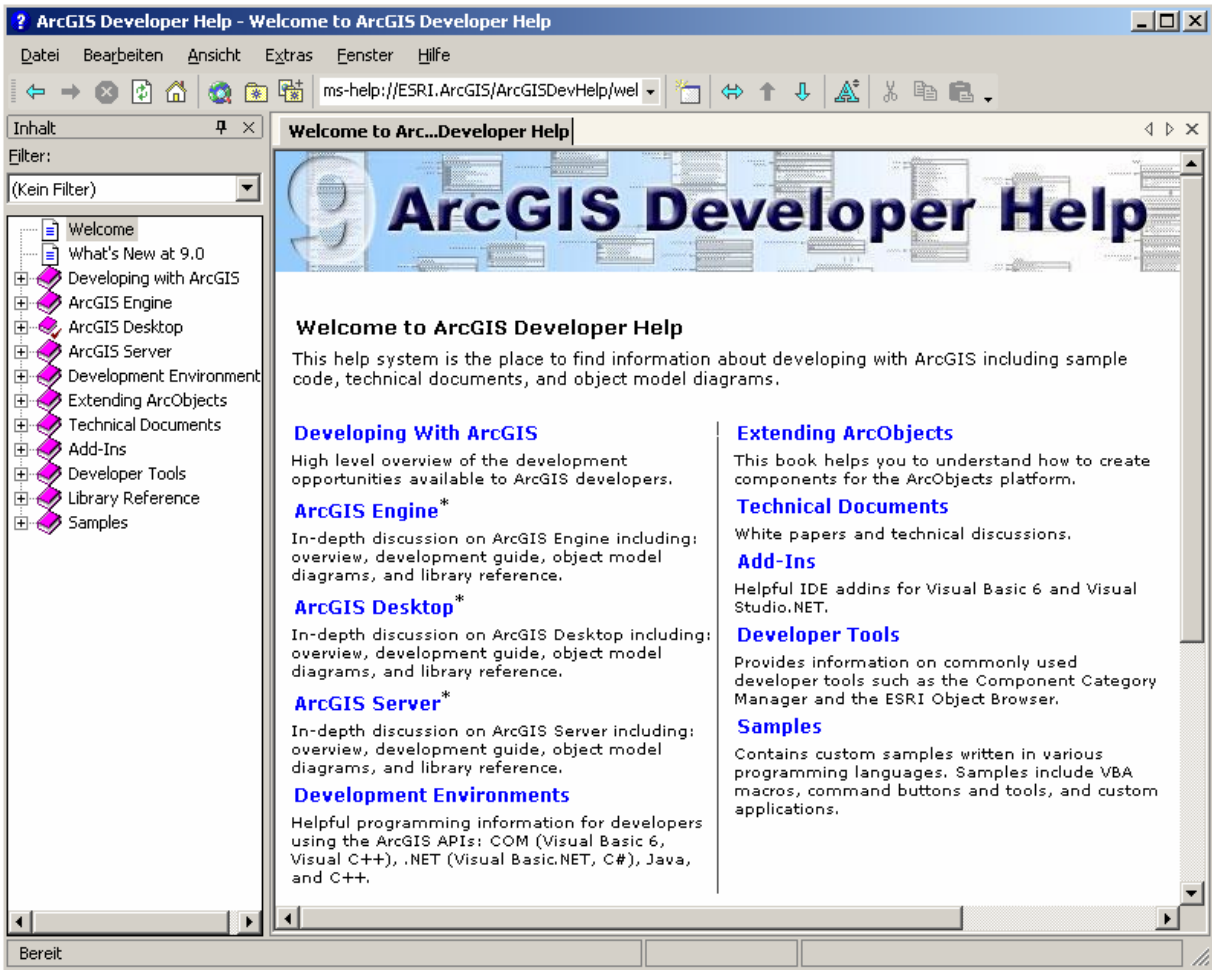

Abb. 26: Hauptfenster ArcGIS Developer Help

Quelle: ms-help://ESRI.ArcGIS/ArcGISDevHelp/welcome.htm, DATUM: 22.03.2005

ArcGIS besitzt eine Vielzahl von Bibliotheken, mit denen man, anhand ihrer zugehörigen Objekte und Modelle, flexible Programmieranforderungen bewältigen kann.
Jede Bibliothek besitzt für ihre Objekte jeweils ein oder mehrere Objekt-Modell-Diagramme.
Die folgende Abbildung zeigt die Verzeichnisstruktur einer Bibliothek und das zugehörige Objekt Modell:

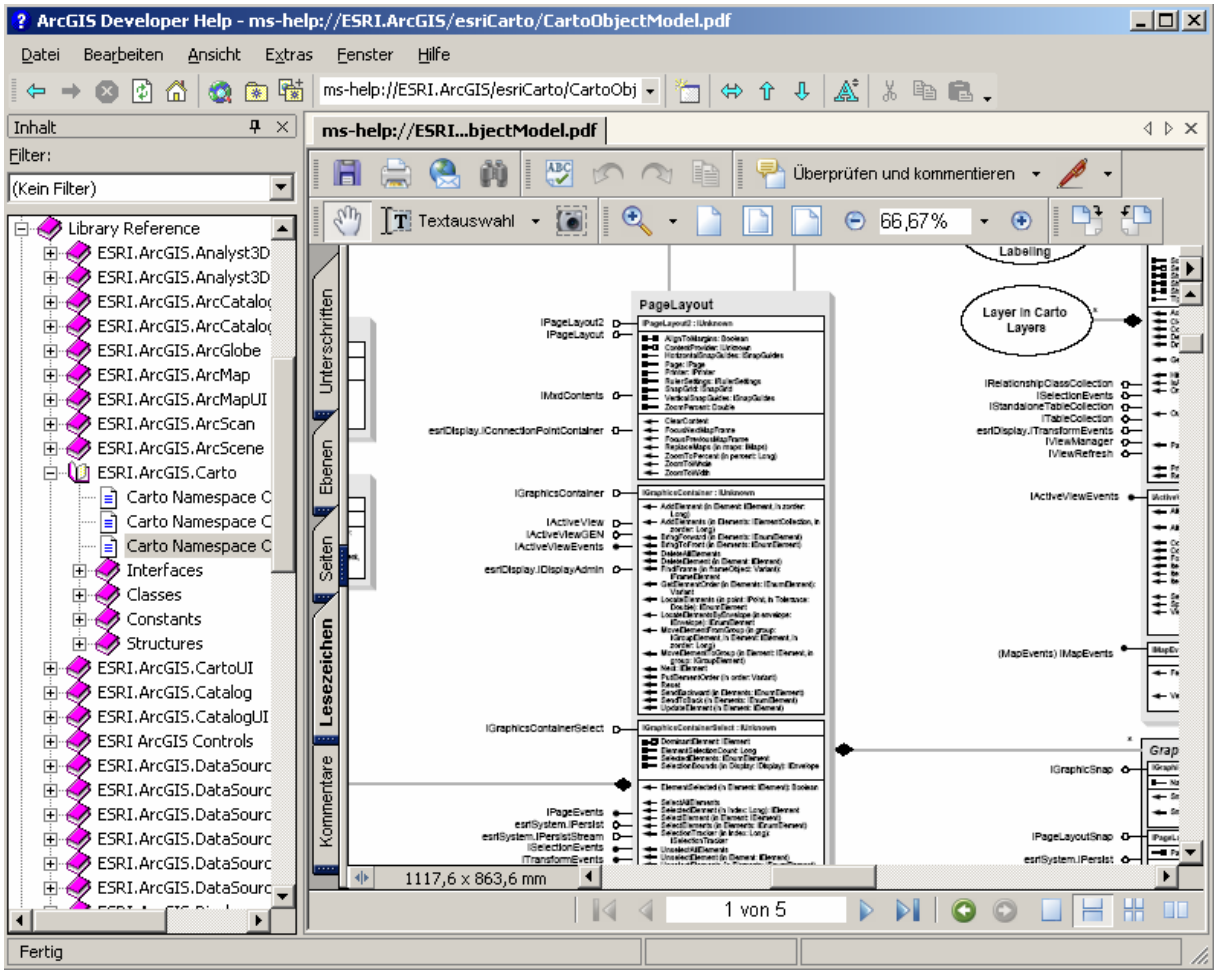

Abb. 27: Objekt Modell der ArcGIS Developer Help

Quelle: ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 1

5.1.1.2 Einarbeitung in die *ArcObjects* Programmierung.

Die ESRI ArcGIS Developer Help beinhaltet auch eine mehr oder weniger kleine Anleitung, bzgl. des Entwickelns von *ArcObjects*.

Diese Abbildung beinhaltet das Verzeichnis der Anleitung für das Entwickeln der ArcObjects:

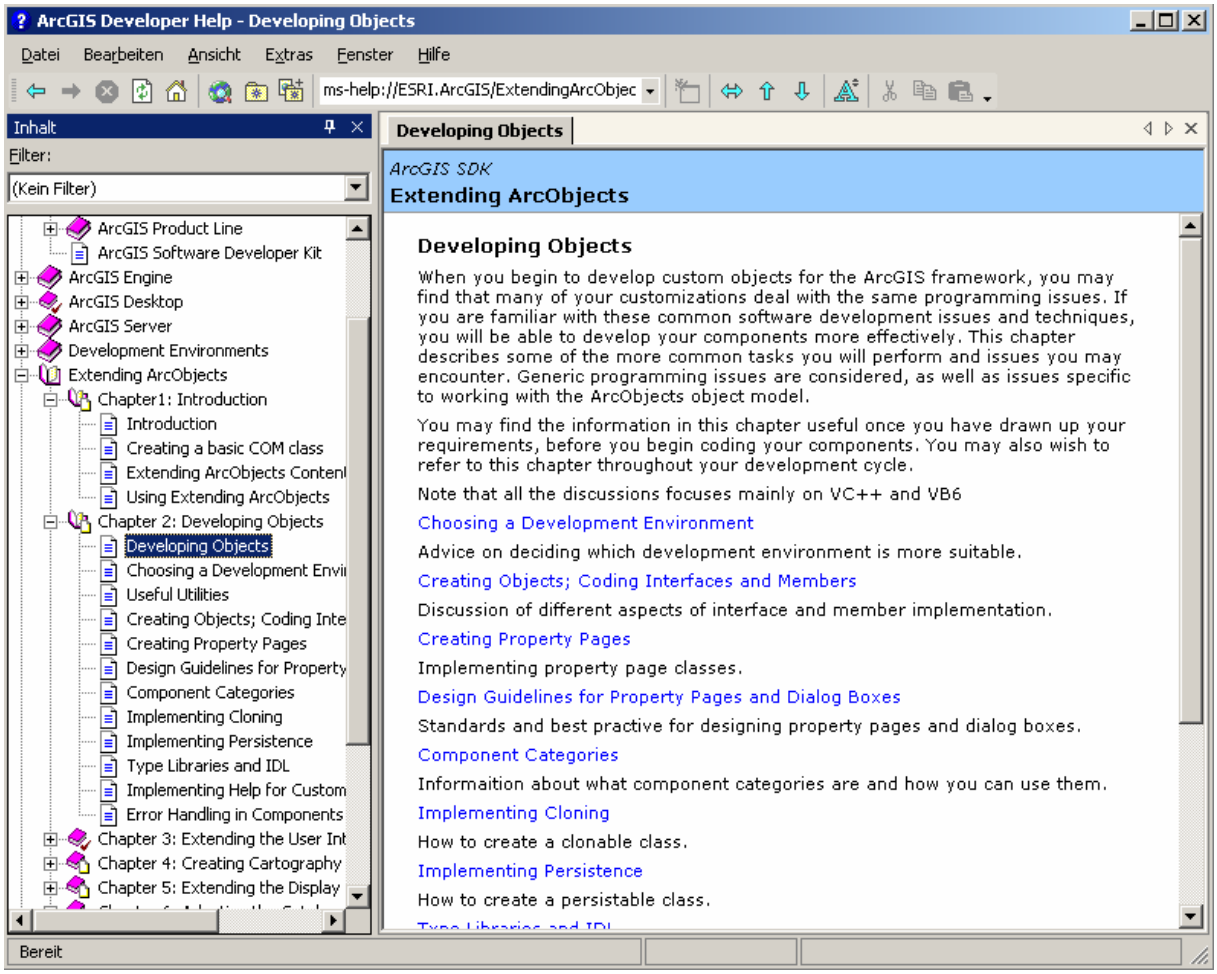

Abb. 28: Anleitung Entwickeln von Objekten der ArcGIS Developer Help

Quelle: ms-help://ESRI.ArcGIS/ExtendingArcObjects/Ch02/DevelopingObjects.htm,
DATUM: 23.03.2005

5.1.2 Test von kleinen Code-Beispielen der ArcGIS Developer Help

Die Developer Help beinhaltet eine große Anzahl von Code-Beispielen, die verschiedene Anwendungsbereiche berühren. Für den Bereich der serverseitigen Programmierung stehen die Beispiele in einer eher dürftigen Anzahl zur Verfügung. Außerdem befinden sich in manchen Beispielen kleine Fehler. Man sollte sich auf jeden Fall mit der VB .NET und VB Programmiersprache auseinander setzen, da viele Beispiele nicht in C# geschrieben sind.
Diese Abbildung zeigt das Verzeichnis der Samples:

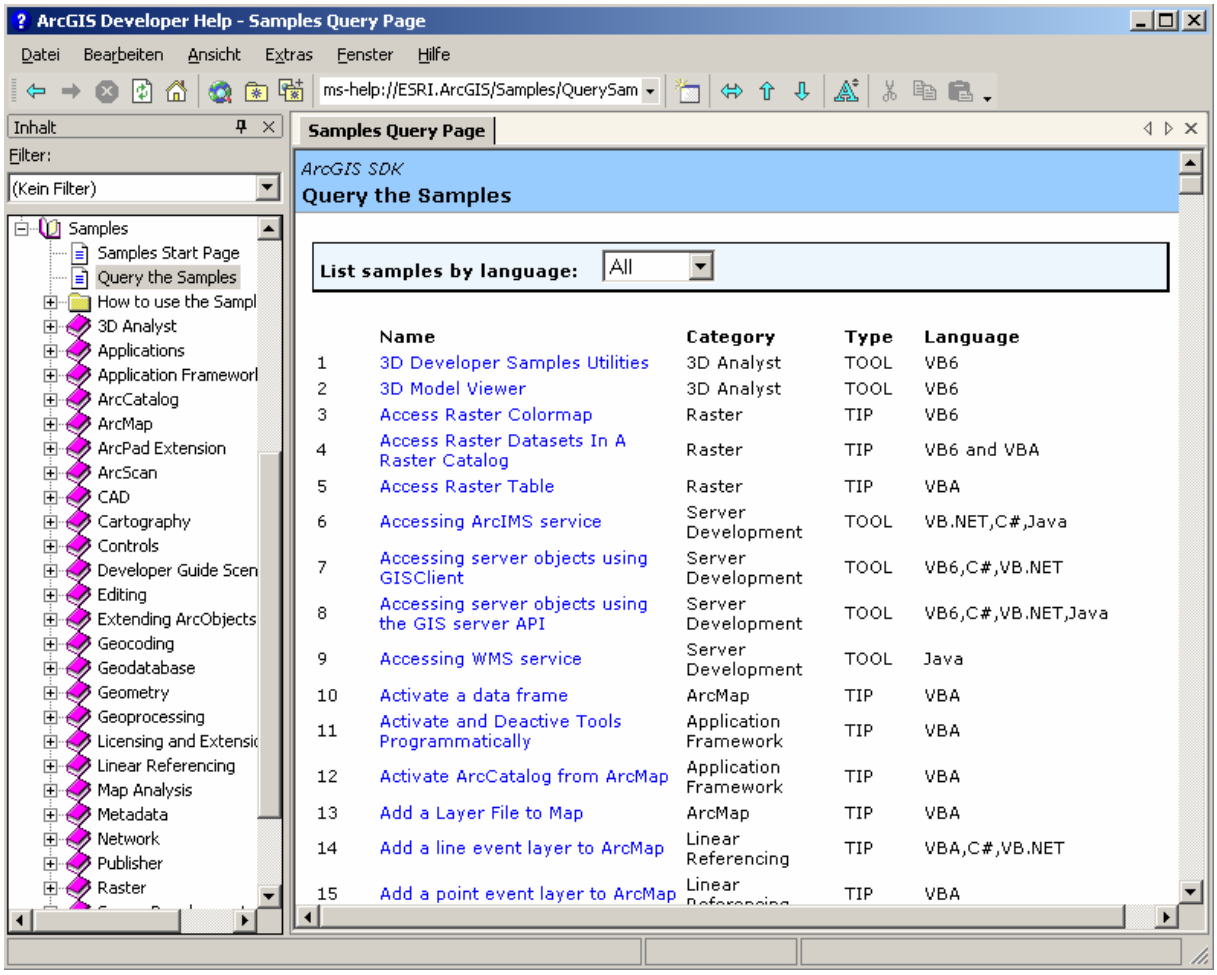

Abb. 29: Anleitung Entwickeln von Objekten der ArcGIS Developer Help

Quelle: ms-help://ESRI.ArcGIS/Samples/QuerySamples.htm, DATUM: 23.03.2005

5.1.2.1 Test der Export-Funktion im ArcMap

Das Beispiel der Export-Funktion im ArcMap ist in C# geschrieben. Der Anwender markiert den zu exportierenden Bereich mit Hilfe eines Rechteckes, das er über die Fläche zieht. Die folgende Abbildung zeigt das Ziehen des Rechteckes:

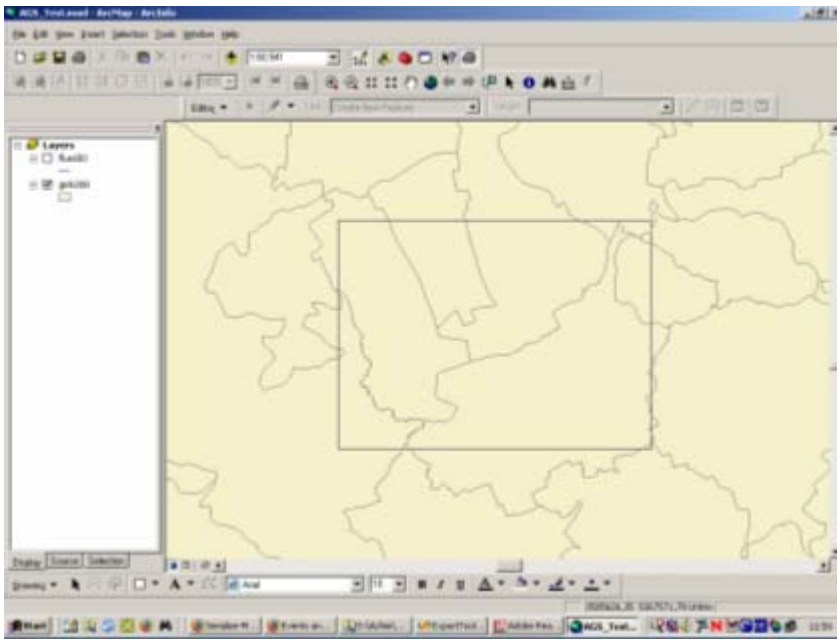

Abb. 30: Export-Funktion ArcMap, BoundingBox

Quelle: Export-Funktion ArcMap, BoundingBox, DATUM: 23.03.2005

Danach öffnet sich ein Dialog-Fenster, in dem man die PDF Datei abspeichern kann:

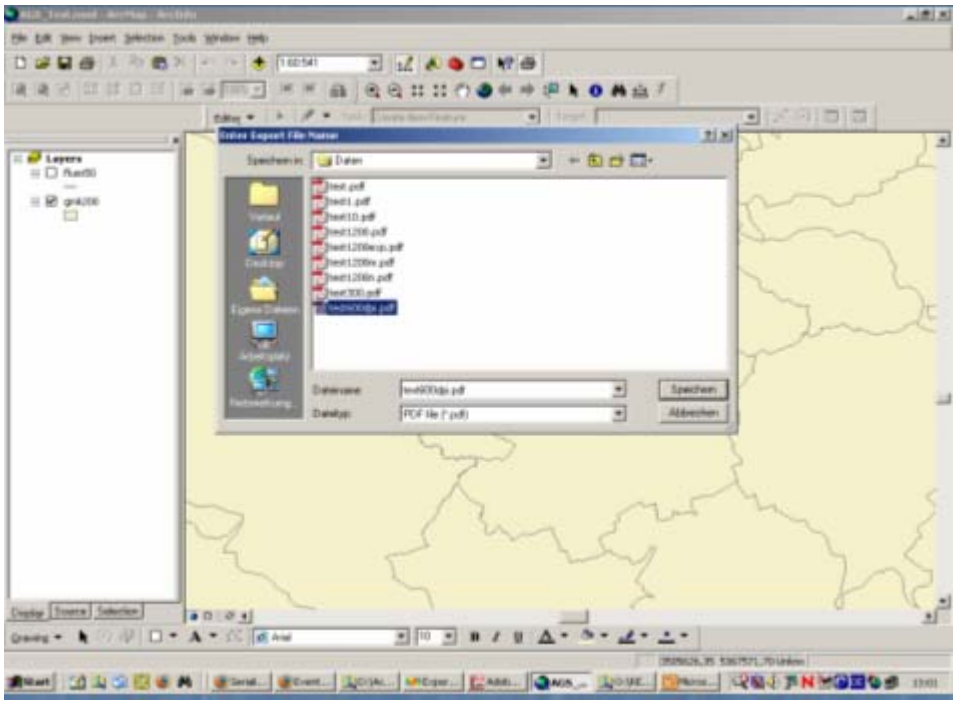

Abb. 31: Export-Funktion ArcMap, Dialog-Fenster

Nach dem Abspeichern des Ausschnittes in eine PDF-Datei, kann man sich diese im Acrobat Reader anschauen und anschließend ausdrucken:

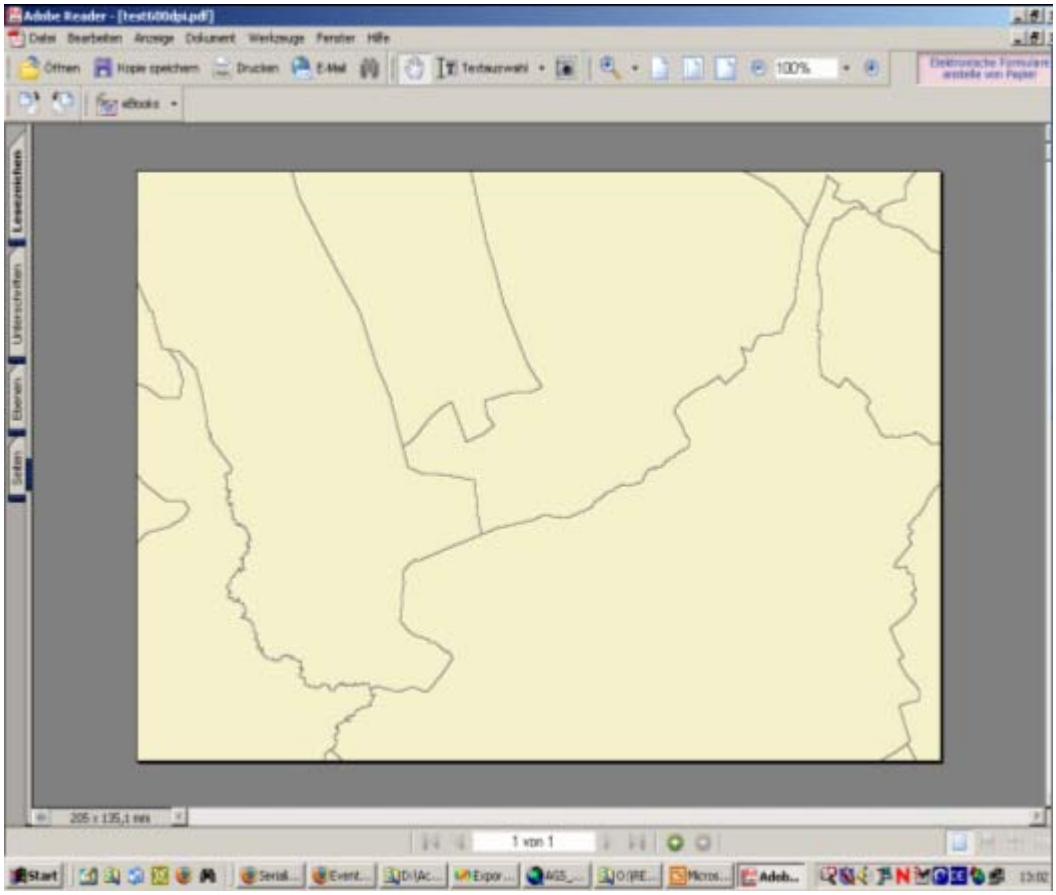

Abb. 32: Ansicht im Acrobat Reader

Quelle: Export-Funktion ArcMap, Acrobat Reader, DATUM: 24.03.2005

5.2 Realisierung und Testen der Funktionen

5.2.1 Das PageLayoutViewer Template des ADF .NET Frameworks

Wie in Kapitel 4 (Abschnitt 4.1) beschrieben, wird in der Umgebung von Visual Studio .NET die Programmierung realisiert. Diese beinhaltet auch nach der Installation des ADF .NET Framework verschiedene Templates. Unter anderem das *PageLayoutViewer Template*, welches in der folgenden Abbildung dargestellt ist:

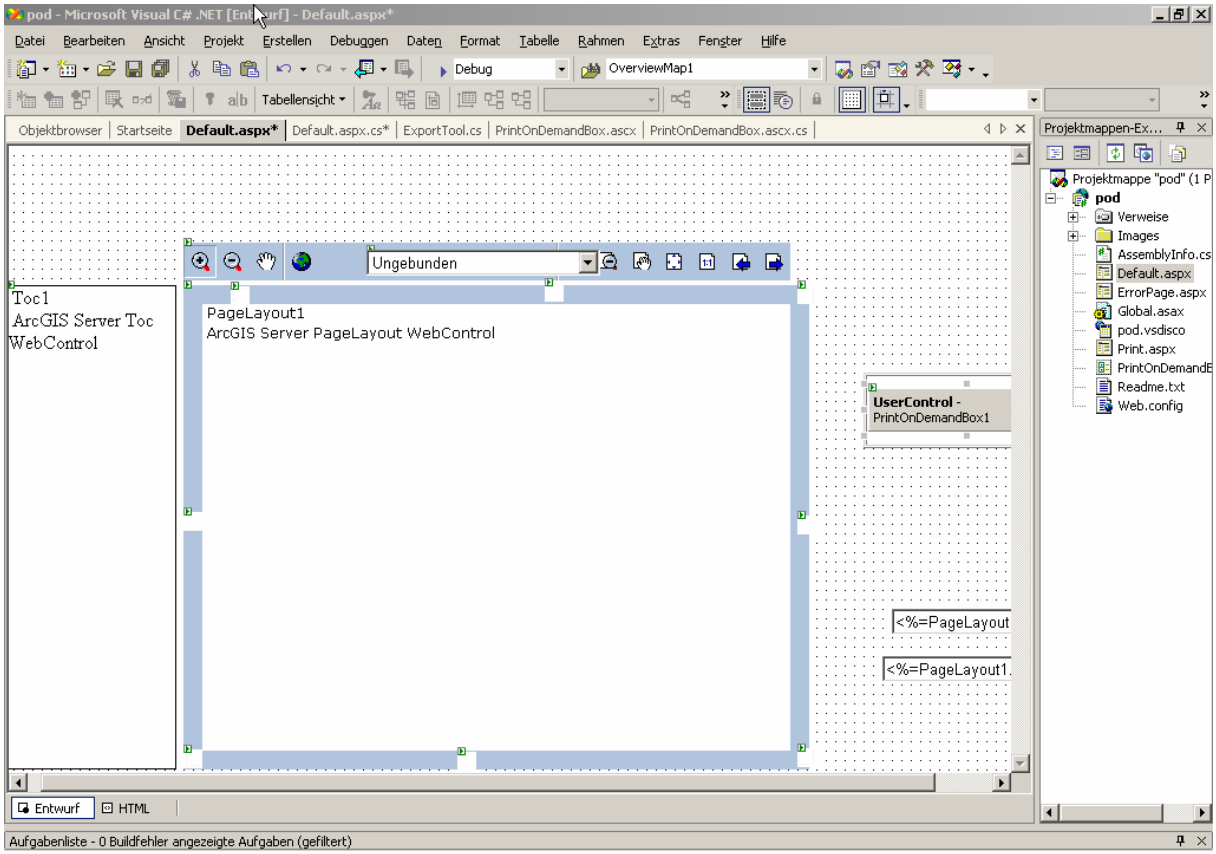

Abb. 33: PageLayoutViewer Template

Quelle: PageLayout Viewer Template des ADF .NET Frameworks, DATUM: 24.03.2005

Im Ansichtsfenster dieses Templates wird bei der Anzeige im Browser ein Karten-Layout angezeigt, das entweder als ArcMap PageLayout-Template (.mxt Datei) vorliegt oder im Map-Viewer des ArcGIS Servers erstellt wurde.

Um den zeitlichen Rahmen der Diplomarbeit nicht zu überschreiten, wird nur auf den PageLayoutViewer eingegangen. der das Bestimmen des Formats, der Auflösung, das Exportieren des Layouts und das Versenden per E-Mail beinhaltet.

Die Funktionen, wie z.B. Maßstab ändern und Seiten-Layout festlegen, werden im Kapitel „Ausblick“ theoretisch behandelt.

5.2.1.1 Einbindung des WebUserControls (Web-Benutzersteuerelement)

Die Erstellung des Layouts und die Einbindung der Funktionen des Web*UserControls* geschieht ebenfalls mit Hilfe von Visual Studio .NET. Der große Vorteil eines solchen Controls ist, dass es mit anderen Komponenten gekoppelt werden kann.
Im Folgenden wird die Entwicklung der einzelnen Komponenten beschrieben.

5.2.1.2 Entwicklung der Funktionen „Export Format" und „Auflösung"

Diese Komponenten beinhalten mehrere Bibliotheken, die in Kapitel 5 näher beschrieben wurden.

Abb. 34: Auflösung und Format des UserControls

Quelle: PrintOnDemandBox.ascx Datei des Projektes, DATUM: 24.03.2005

Diese Komponenten beinhalten ein DropDown-Menü, welche dem Benutzer die Auswahl lässt zwischen 96 dpi, 300 dpi und 600 dpi zu wählen, sowie das jeweilige Export-Format. Die Druckqualität steigt je höher der Wert der Auflösung ist. Der Nachteil besteht aber darin, dass sich durch die Erhöhung des Wertes auch die Dateigröße erhöht. Im folgenden Abschnitt wird näher auf den Ablauf des Codes eingegangen.

Die Export-Funktion setzt sich aus dem Verbindungen zwischen den verschiedenen Bibliotheken und ihren Objekten zusammen. Diese Funktion beinhaltet die Carto Bibliothek, Display Bibliothek, Output Bibliothek und das *WebControls* Namespace.
Folgende Schritte werden bei dieser Funktion ausgeführt:

- Der Focus wird auf das aktive *PageLayout* Objekt gerichtet
- Erstellung einer neuen Export-Klasse und Bestimmung des Output-Verzeichnisses
- Festlegung der Auflösung
- Bestimmung des Objektes für das Abfangen der Eckpunkt-Koordinaten
- Abfangen eines Rechteckes mit den 4 Eckpunkten

- Erstellung eines neuen Rechteckes und Setzen der Koordinaten
- Initialisierung des Exporters
- Übergabe der Parameter an ein Ausgabegerät
- Beenden des Exporters

Folgende Abb. zeigt die komplette Export-Funktion mit den Kommentaren:

```
if (ExportType.SelectedValue = "PDF")
{
    // Focus wird auf das aktive PageLayout Objekt gerichtet
    IActiveView pActiveView = webPageLayout.PageLayout as IActiveView;

    // Erstellung einer neuen Export-Klasse und Bestimmung des Output-Verzeichnisses
    IExport pExport = m_ctx.CreateObject("esriOutput.ExportPDF") as IExport;
    pExport.ExportFileName = "F:\\Achim\\outputfolder\\" + session_id + ".pdf";

    // Festlegung der Auflösung
    int iScreenResolution = 96;
    pExport.Resolution = imageRes;

    // Bestimmung des Objektes für das Abfangen der Eckpunkt-Koordinaten
    IDisplayTransformation pDT = pActiveView.ScreenDisplay.DisplayTransformation as IDisplayTransformation;
    IEnvelope pEnv = pDT.VisibleBounds as IEnvelope;

  // Abfangen eines Rechteckes mit den 4 Eckpunkten
  tagRECT exportRECT;
  exportRECT.left = 0;
  exportRECT.top = 0;
  exportRECT.right = pActiveView.ExportFrame.right * (imageRes / iScreenResolution);
  exportRECT.bottom = pActiveView.ExportFrame.bottom * (imageRes / iScreenResolution);

  // Erstellung eines neuen Rechteckes und setzen der Koordinaten
  IEnvelope pPixelBoundsEnv;

  pPixelBoundsEnv = m_ctx.CreateObject("esriGeometry.Envelope") as IEnvelope;
  webPageLayout.ManageLifetime(pPixelBoundsEnv);

  pPixelBoundsEnv.PutCoords(exportRECT.left, exportRECT.top, exportRECT.right, exportRECT.bottom);
  pExport.PixelBounds = pPixelBoundsEnv;

  // Initialisiert den Exporter
  int hDC;
  hDC = pExport.StartExporting();

  // Ermöglicht den Benutzer, mit Hilfe einer Taste den Export zu unterbrechen
  ITrackCancel pCancel = new CancelTrackerClass() as ITrackCancel;


 // Übergabe der Parameter an ein Ausgabegerät
 pActiveView.Output(hDC, imageRes, ref exportRECT, pEnv, pCancel);

  // Beendet den Exporter
  pExport.FinishExporting();
  pExport.Cleanup();
```

Abb. 35: Code Export-Funktion

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 181 - 223

5.2.1.2.1 Ergebnis

Der Benutzer wählt die Auflösung und das Format, damit durch den Sende-Button die Datei erstellt werden kann.

Die folgende Abb. zeigt den PageLayoutViewer mit dem „Print on demand“ Steuerelement:

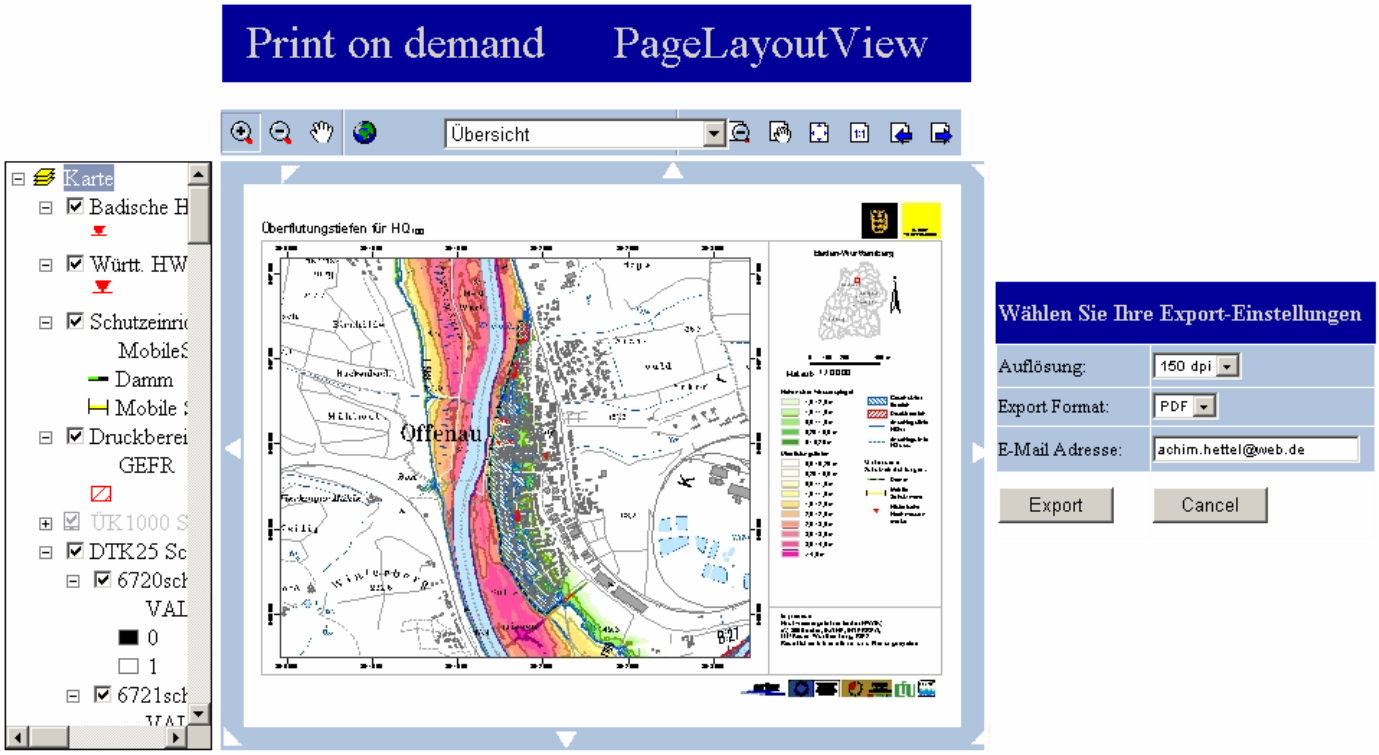

Abb. 36: PageLayoutViewer mit Exportelement

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 181 - 223

Mit demselben Ausschnitt wurde auch in ein JPEG-Format exportiert. Ein sehr großer Nachteil besteht darin, dass die Datei um ein vielfaches größer wird als bei einem Export in ein PDF-Format. Hiermit stellt sich natürlich die Frage, ob man nicht generell das PDF-Format als Export-Format benutzt.

Die exportierte Datei wird in einem Output-Verzeichnis abgelegt, die der Benutzer dann über eine E-Mail Message herrunterladen oder abrufen kann.
Folgende Abb. zeigt die gespeicherte Datei im Output-Verzeichnis:

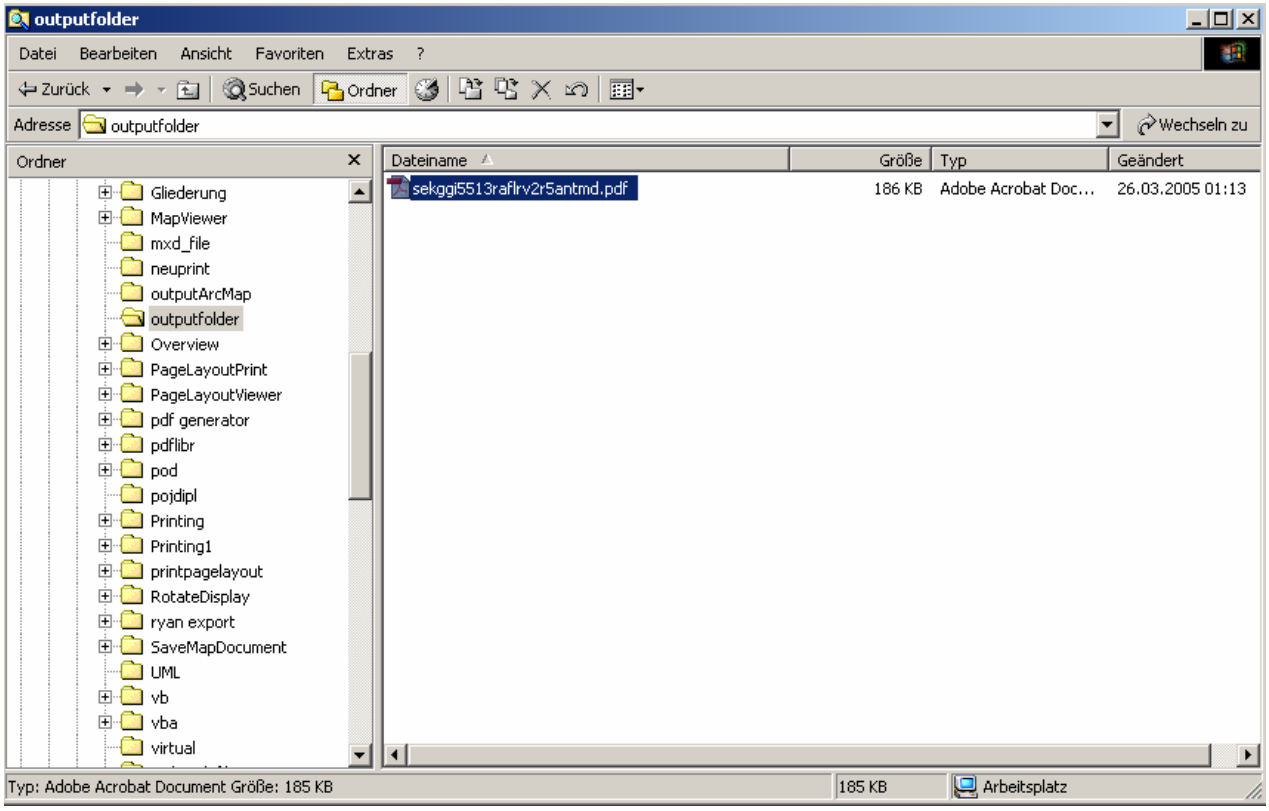

Abb. 37: Datei-Verzeichnis der Ausgabe-Datei

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 181 - 223

5.2.1.3 Entwicklung der Funktion „E-Mail Adresse"

Diese Komponente stellt ein Eingabefeld dar, in dem der Benutzer seine E-Mail Adresse eingeben kann.
Der Benutzer soll die Möglichkeit haben, das von ihm gewählte Layout per E-Mail auf sein Account zugeschickt zu bekommen. Dort hat er dann die Möglichkeit, die Datei herunterzuladen.

Folgende Abb. zeigt nochmals das Steuerelement mit dem Eingabefeld:

Abb. 38: E-Mail Komponente des UserControls

Quelle: PrintOnDemandBox.ascx Datei des Projektes, DATUM: 25.03.2005

Wie in Kapitel 5 (Abschnitt 5.9) beschrieben beinhaltet diese Komponente den Microsoft .NET Framework System.Web.Mail-Namespace. Anhand der dazugehörigen Objekte wird dem Benutzer die Datei auf seinen Account geschickt.
Die folgende Abb. des Codes zeigt die Sende-Methode:

```
            url = webPageLayout.Export(EXPORT_IMAGE_FORMAT);
        }
    }
    //E-Mail Message an Benutzer
    SendMail(m_MailToAddress, String.Format("Ihre Datei können Sie downloaden unter:<a href='{0}'> {0}</a>.", url));
```

Abb. 39: Code Sende Methode

Quelle: pod.WebUserControl.PrintOndemandBox.ascx, Zeile 230 - 234

5.2.1.3.1 Ergebnis

Die folgende Abb. zeigt die E-Mail Message an den Benutzer:

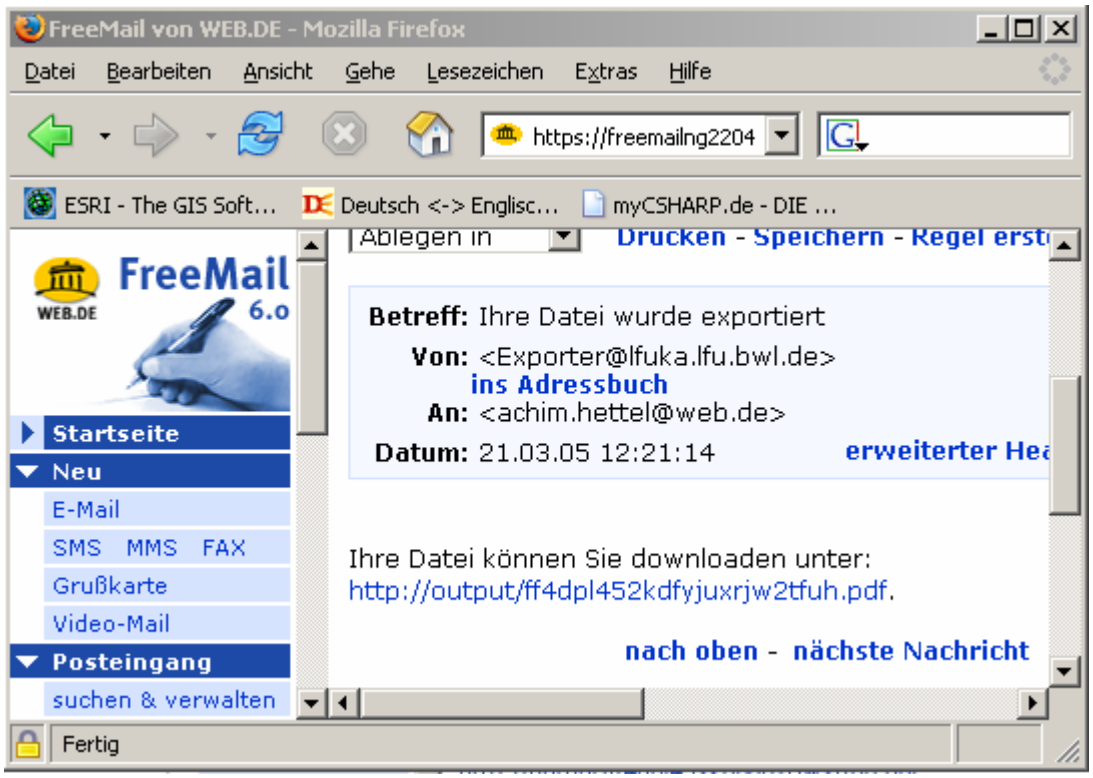

Abb. 40: E-Mail Account mit Nachricht

5.2.1.4 Die Konfiguration der XML Datei

Wie in Kapitel 6 (Abschnitt 6.1.1) beschrieben, besitzt dieses Projekt auch eine XML-Datei. Diese Datei wurde als Web.config (Konfigurationsdatei) angelegt.

Relevant für das Projekt, ist die Impersonation (Personifizierung) und die Verwaltung der Sessions.

Folgende Abb. zeigt den Code der Personifizierung:

```
<identity impersonate="true" userName="" password="" />
```

Abb. 41: XML Code Impersonation

Quelle: pod.WebUserControl.Web.config, Zeile 35 - 42

Hier wird der Benutzername und das Passwort, welche verschlüsselt sind, innerhalb des Servers angesprochen. Dies dient zur Sicherheit des Servers.

Diese Abb. zeigt den Code der Session-Verarbeitung:

```
<sessionState
        mode="InProc"
        stateConnectionString="tcpip=127.0.0.1:42424"
        sqlConnectionString="data source=127.0.0.1;user id=sa;password="
        cookieless="false"
        timeout="20"
        stateNetworkTimeout="500"
/>
```

Abb. 42: XML Code SessionState

Quelle: pod.WebUserControl.Web.config, Zeile 57 - 64

Wichtig ist die *timeout* Eigenschaft, die eine Zeit in Minuten referenziert. Die gespeicherte Export-Datei wird nach Ablauf dieser Zeit wieder gelöscht. Dies soll verhindern, dass der Server durch zu viel gespeicherte Dateien seine Stabilität verliert.

5.3 Besonderheiten und Probleme

5.3.1 Einbindung des Impersonation

Im PageLayout Template des ADF.NET Frameworks ist ein *Impersonation* Control eingebaut, das nicht funktioniert. Die Anwendung konnte dadurch nicht gestartet werden.
Durch das Entfernen des Controls, das Aktivieren einer Methode im Code und durch das Zuweisen in der Web.config Datei konnte die Anwendung schließlich doch gestartet werden.

5.3.2 Literatur und fehlerhafte Codes

In Bezug auf das Thema dieser Diplomarbeit gibt es fast keine Beispiele.
Andere Beispiele, die ein wenig mehr Aussagekraft besitzen, sind in der Programmiersprache VB geschrieben, und zudem nicht für Server-Anwendungen. Somit musste der Verfasser sich zusätzlich noch in die Programmiersprache VB einarbeiten. Einige Beispiele beinhalten auch kleinere Fehler im Code.
Das ArcGIS Server Forum von ESRI ist kein Forum, das regelmäßig von vielen Usern besucht wird.
Es besitzt daher nur eine kleine Anzahl von Beiträgen und auf Fragen des Verfassers wurde oft nicht geantwortet.
Außer der ESRI Help gibt es keine Literatur, die sich mit der Thematik der Diplomarbeit auch nur annähernd befasst.

6 Ausblick

6.1 Konzept

Das Projekt könnte mit einem MapViewer beginnen, welcher ein Kartenobjekt beinhaltet. Diese Seite enthält zugleich ein Steuerelement, das dem Benutzer ermöglicht, die verschiedenen Papierformate und den Maßstab auszuwählen. Nach dem Auswählen soll der Benutzer den Button „Layout-Ansicht“ drücken, um die Karte im PageLayoutView (siehe Kapitel 5, Abschnitt 5.2.1.2.1) anzeigen zu lassen. Die Legende soll während des Wechselns, zwischen den zwei Ansichten automatisch generiert werden. Untersuchungsfähig wäre noch die Möglichkeit, die Legende als Template einzubinden oder dynamisch vom Benutzer erstellen lassen.
Die folgende Abb. stellt den MapViewer dar:

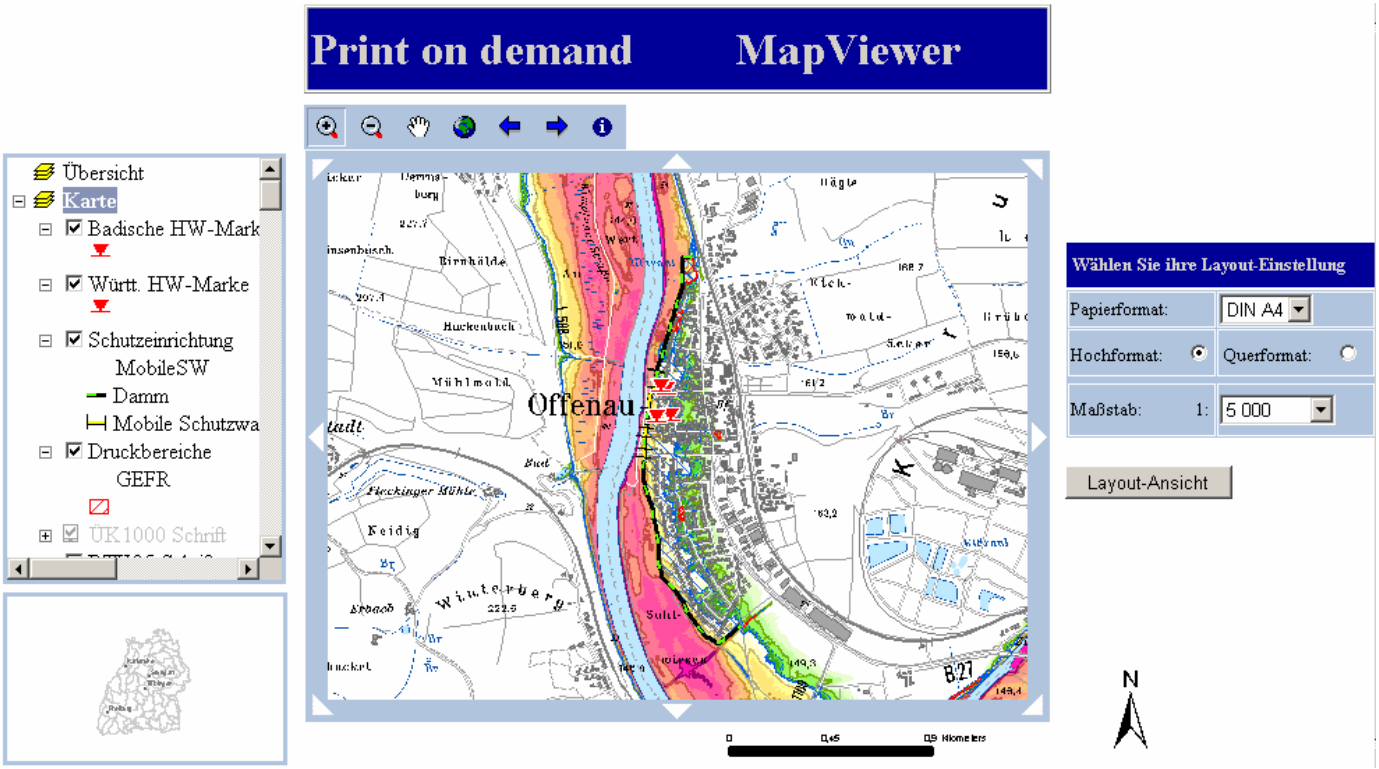

Abb. 43: MapViewer mit Layoutelement

Quelle: podmapview.WebUserControl.pod.ascx, Zeile 175 - 215

Die einzelnen Komponenten des Steuerelementes werden in den folgenden Abschnitten näher beschrieben.

6.1.1 Entwicklung der Komponente „Papier-Format“

6.1.1.1 Beschreibung

Um den Funktionsumfang flexibler gestalten zu können, wäre die Auswahl des Papier-Formats durch den Benutzer eine sinnvolle Erweiterungsmöglichkeit.
Die folgende Abb. zeigt das Benutzersteuerelement mit der Auswahl des Papierformates:

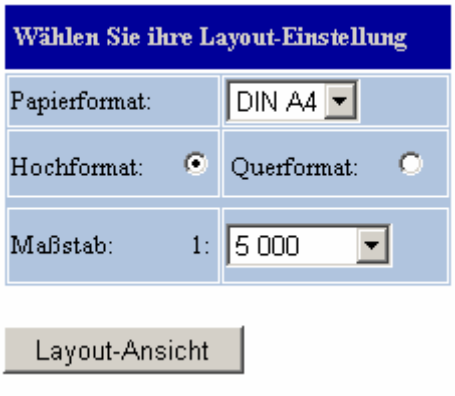

Abb. 44: UserControl Layout

Quelle: PrintOnDemandBox.ascx Datei des Projektes, DATUM: 26.03.2005

Die möglichen relevanten Klassen, Methoden und Eigenschaften wurden in Kapitel 4 (Abschnitt 4.4.2.3) näher beschrieben.
Da es Kartenwerke gibt, die aufgrund der Maßstäbe keinen Ausdruck auf bestimmte Papierformate zulassen, wäre hier noch die Möglichkeit anhand einer XML-Datei, bestimmte Papierformate zu deaktivieren.

6.1.2 Entwicklung der Komponente „Maßstab“

6.1.2.1 Beschreibung

Bei Auswahl eines Maßstabes soll die Karte automatisch auf den gewünschten Wert mittig gezoomt werden.
Diese Komponente beinhaltet Interfaces der Carto Bibliothek, Geometry Bibliothek und des WebControls Namespaces.
Die folgende Auflistung zeigt die möglichen Interfaces:

- **ICenterAndScale** (Carto)
- **IMapDescription** (Carto)
- **IPoint** (Geometry)
- **IMapArea** (Carto)

In den folgenden Abschnitten wird näher auf die möglichen relevanten Interfaces eingegangen.

6.1.2.1.1 Das *IMapDescription* Interface der Carto Bibliothek

Dieses Interface verwaltet die Eigenschaften des Datenrahmens einer Karte.

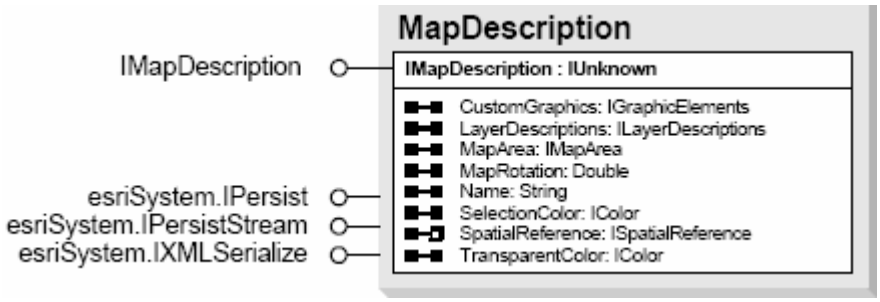

Abb. 45: IMapDescription Interface

Quelle ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 4

Eine relevante Eigenschaft beinhaltet dieses Interface:

- Die ***MapArea*** Eigenschaft
  Diese Eigenschaft beinhaltet das Gebiet einer Kartenansicht und stellt einen Verweis auf das *IMapArea* Interface her.

Folgender Codeausschnitt soll dies verdeutlichen:

```
IMapDescription mapDescription = webMap.MapDescription;
IMapArea mapArea = mapDescription.MapArea;
```

Im 1. Abschnitt wird der Datenrahmen eines WebMap-Objektes bestimmt.
Im 2. Abschnitt wird das Gebiet eines Datenrahmens bestimmt und einen Verweis auf das IMapArea Interface geführt.

6.1.2.1.2 Das *IMapArea* Interface der Carto Bibliothek

Dieses Interface gibt den Wert des Umfanges der Karte zurück, welcher von den Eigenschaften des *IEnvelope* Interfaces abhängig ist.

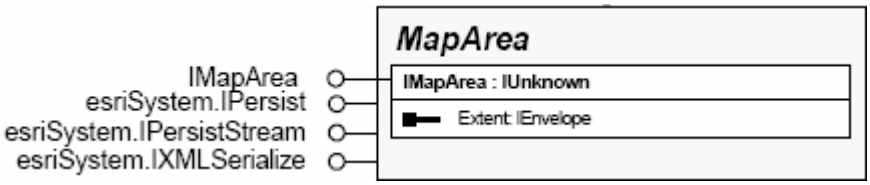

Abb. 46: IMapArea Interface

Quelle ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 4

Dieses Interface besitzt nur eine Eigenschaft:

- Die ***Extent*** Eigenschaft
  Gibt den Wert des Umfanges der Karte zurück.

Folgender Codeausschnitt verdeutlicht das Abfangen des Wertes:

IMapArea mapArea = mapDescription.**MapArea**;

centerPoint.X = mapArea.**Extent**.XMin+(mapArea.**Extent**.XMax- mapArea.**Extent**.XMin) / 2;

centerPoint.Y = mapArea.**Extent**.YMin+ (mapArea.**Extent**.YMax- mapArea.**Extent**.YMin) / 2;

6.1.2.1.3 Das *IPoint* Interface der Geometry Bibliothek

Dieses Interface spezifiziert einen Punkt auf dem ArcMap Dokument mit den X und Y-Werten.

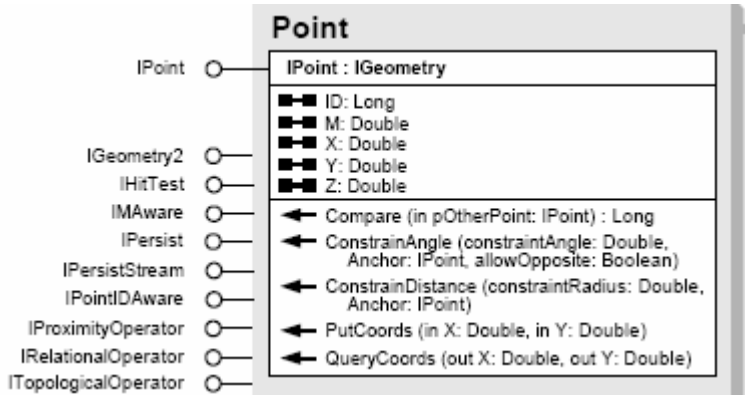

Abb. 47: IPoint Interface

Quelle ms-help://ESRI.ArcGIS/esriGeometry/GeometryObjectModel.pdf, Seite 1

Dieses Interface beinhaltet zwei relevante Eigenschaften:

- Die ***X*** Eigenschaft
  Bestimmt den X-Wert des Punktes
- Die ***Y*** Eigenschaft
  Bestimmt den Y-Wert des Punktes

Ziel ist es, den Punkt auf die Mitte der Kartenansicht zu fokusieren, um ein mittiges zoomen zu gewährleisten.
Der folgende Codeausschnitt beinhaltet die Zentrierung des Punktes:

```
IPoint centerPoint = serverContext.CreateObject("esriGeometry.Point") as IPoint;

centerPoint.X = mapArea.Extent.XMin+ (mapArea.Extent.XM - mapArea.Extent.XMin) / 2;

centerPoint.Y = mapArea.Extent.YMin+ (mapArea.Extent.YMax - mapArea.Extent.YMin) / 2;
```

Im 1. Abschnitt wird ein neues Objekt erstellt.
Im 2. Abschnitt wird der X-Wert anhand des Kartenmittelpunktes bestimmt
Im 3. Abschnitt wird der Y-Wert anhand des Kartenmittelpunktes bestimmt

6.1.2.1.4 Das *ICenterAndScale* Interface der Carto Bibliothek

Das Interface manipuliert die Ausdehnung des Kartenobjektes.
Die folgende Abb. zeigt das Interface:

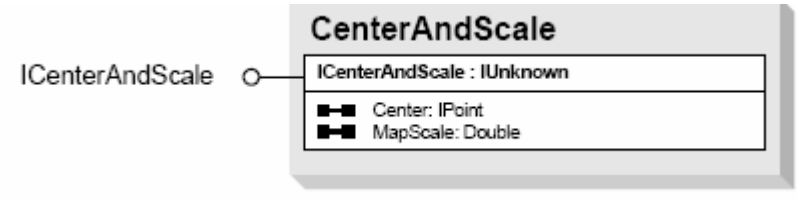

Abb. 48: ICenterAndScale Interface

Quelle: ms ms-help://ESRI.ArcGIS/esriCarto/CartoObjectModel.pdf, Seite 4

Das Interface besitzt zwei Eigenschaften, die dafür verantwortlich sind:

- Die ***Center*** Eigenschaft
  Beinhaltet das Zentrum der Karte, mit einem Verweis zum *IPoint* Interface.
- Die ***MapScale*** Eigenschaft
  Beinhaltet den Wert der Skalierung der Karte.

Der folgende Codeausschnitt soll dies noch einmal verdeutlichen:

```
ICenterAndScale centerScale = serverContext.CreateObject("esriCarto.CenterAndScale");

centerScale.Center = centerPoint;

centerScale.MapScale = Convert.ToInt32(DDScale1.SelectedItem.Value);
```

Im 1. Abschnitt wird ein neues Objekt erzeugt.
Im 2. Abschnitt die Center-Eigenschaft auf das IPoint Interface referenziert.
Im 3. Abschnitt wird der Skalierungswert des DropDown-Menüs ausgelesen.

6.1.2.1.5 Die Methode im Überblick

Der folgende Code beinhaltet eine mögliche Methode:

```
private void DDScale1_SelectedIndexChanged(object sender, System.EventArgs e)
 {
  using (WebMap webMap = Map1.CreateWebMap())
  {

    IMapServer mapServer = webMap.MapServer;
    IServerContext serverContext = webMap.ServerContext;
    IMapDescription mapDescription = webMap.MapDescription;

    IMapArea mapArea = mapDescription.MapArea;

IPoint centerPoint = serverContext.CreateObject("esriGeometry.Point") as IPoint;
centerPoint.X = mapArea.Extent.XMin + (mapArea.Extent.XMax - mapArea.Extent.XMin) / 2;
centerPoint.Y = mapArea.Extent.YMin + (mapArea.Extent.YMax - mapArea.Extent.YMin) / 2;

ICenterAndScale centerScale = serverContext.CreateObject("esriCarto.CenterAndScale")

centerScale.Center = centerPoint;
centerScale.MapScale = Convert.ToInt32(DDScale1.SelectedItem.Value); //100, 1000, 5000, 10000

webMap.MapDescription.MapArea = centerScale as ESRI.ArcGIS.Carto.IMapArea;
webMap.Refresh();

}
}
```

6.1.3 Die Generierung der Legende

6.1.3.1 Überblick über die Programmierung

Wie in Kapitel 6 (Abschnitt 6.1) beschrieben, soll die Legende während des Wechselns zwischen der Kartenansicht und der Layoutansicht generiert werden.
Auch hier stellt das ADF bestimmte Interfaces bereit, um die Legende in Bezug auf die verschiedenen Layer einer Karte, im Layout anzuzeigen.
Folgende Interfaces werden hierfür benötigt:

- **IPageLayout** (Carto)
- **IMapFrame** (Carto)
- **IMapSurroundFrame** (Carto)
- **ILegend** (Carto)
- **ILegendFormat** (Carto)
- **IElement** (Carto)
- **IEnvelope** (Geometry)

Die folgenden Abschnitte befassen sich grob mit den verschiedenen Interfaces.
Interfaces, die in früheren Kapiteln beschrieben wurden, werden nicht mehr beschrieben.

6.1.3.1.1 Das *IMapFrame* Interface der Carto Bibliothek

Der Hauptzweck dieser Schnittstelle ist, dem Entwickler Zugang zu dem Kartenobjekt geben, das innerhalb eines Frames liegt.

6.1.3.1.2 Das *IMapSurroundFrame* Interface der Carto Bibliothek

Diese Schnittstelle beinhaltet den Zugriff auf die Elemente außerhalb des Frames (z.B. Nordpfeil, Legende, Maßstabsleiste).

6.1.3.1.3 Das *ILegend* Interface der Carto Bibliothek

Die Aufgabe dieser Schnittstelle beinhaltet die Eigenschaften und Methoden einer Legende.

6.1.3.1.4 Das *ILegendFormat* Interface der Carto Bibliothek

Mit dieser Schnittstelle werden die Formateigenschaften der Legende verwaltet (z.B. Höhe, Breite, Abstände zwischen den Elementen usw.).

6.1.3.1.5 Das IElement Interface der Carto Bibliothek

Die Elemente (z.B. Polygone, Linien, Text usw.) einer Legende werden mit dieser Schnittstelle verwaltet.

6.1.3.2 Möglicher Ablauf der Generierung einer Legende

Ohne auf die Einzelheiten einzugehen, wird nachfolgend ein möglicher Ablauf der Generierung einer Legende aufgezeigt:

1. Fokus wird auf das Kartenobjekt referenziert (**IPageLayout**, **IMapFrame**))
2. Neuer Rahmen und dessen Größe wird definiert (**IEnvelope**)
3. Legende wird erzeugt (**ILegend**)
4. Format der Legende und ihrer Elemente wird festgelegt (**ILegendFormat**, **IElement**)
5. Referenzierung der Legende und des Rahmens
6. Einfügen in das PageLayout ((**IMapSurroundFrame**)

6.1.4 Einbindung eines Webservices

Ein Webservice dient dazu, Datenbankabfragen und Verbindungen auszulesen.
Webservice sind plattformunabhängig, da auf den Internet Standard XML und HTTP aufbauen. Bei dieser Abbildung handelt es sich um einen Webservice, der auf eine Anfrage hin die Bounding-Box Daten aus einer Access-Datenbank ausliest und zurückschickt.

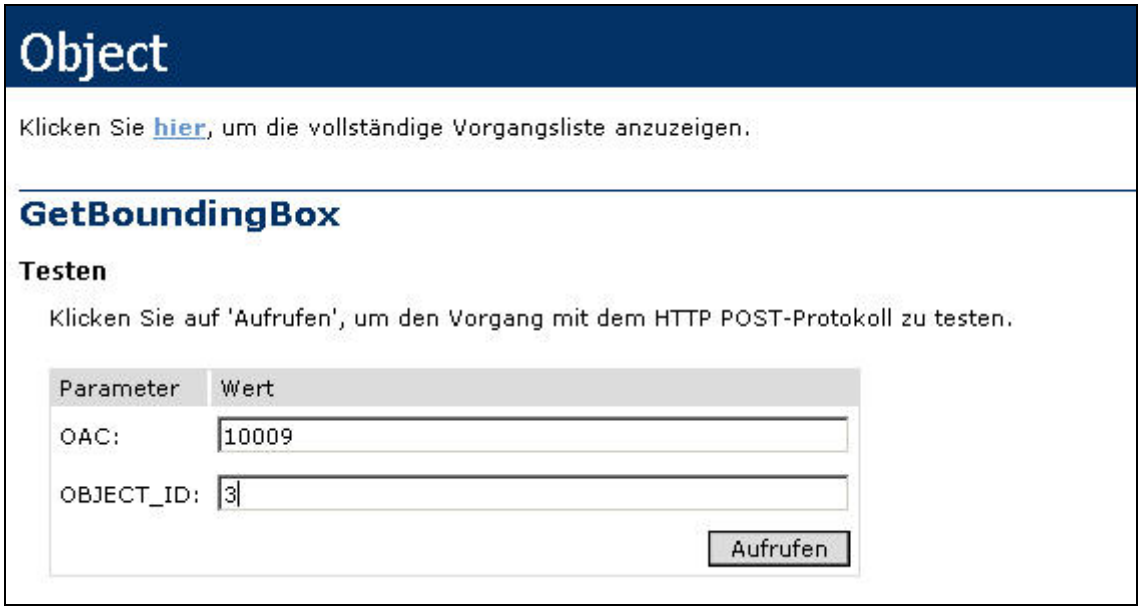

Abb. 49: WebService

Quelle: Diplomarbeit von Michael Neppl, WS 2004 / 2005, LfU Karlsruhe

Anhand eines XML-Dokumentes werden dann die vier Eckpunkte der Boundingbox zurückgeliefert und im Ansichtsfenster als Kartenausschnitt angezeigt. Folgender Codeausschnitt zeigt das Auslesen der vier Eckpunkte in der XML-Datei:

```
<?xml version="1.0" enconding="utf-8" ?>
<BoundingBox xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2002/XMLSchema-instance"
 xmlns="http://tempuri.org/">
 <LL_EASTING>3592817</LL_EASTING>
 <LL_NORTHING>5973302</LL_NORTHING>
 <UR_EASTING>3650266</UR_EASTING>
 <UR_NORTHING>6047117</UR_NORTHING>
</BoundingBox>
```

Diesen Webservice könnte man mit der “Print-on-demand” Funktion koppeln, um eine höhere Flexibilität der Anwendung zu gewährleisten.

7 Zusammenfassung

Durch das stetige Wachsen der Benutzer im Internet, wächst auch zunehmend der Anspruch darauf, dass sich die Art der Informationsbeschaffung immer auf gleicher Höhe mit der aktuellen Technologie befindet. Im Bereich der Geographischen Informationssysteme (GIS) wird erst seit wenigen Jahren Geodaten und GIS-Funktionen im Internet angeboten. Der ArcGIS 9 Server der Firma ESRI ist nur ein Beispiel für die rasante Entwicklung der WebGIS-Anwendungen in den letzten Jahren. Dieser Server steht für eine neue Generation von Map Servern zuzüglich einer breiten Palette an Benutzer-Funktionalitäten.

Das Ziel der Diplomarbeit war, auf Basis dieses Servers eine Export –und Druck-Funktion zu entwickeln, die dem Anwender verschiedene Auswahlmöglichkeiten lässt. Als Service sollte dem Anwender die exportierte Datei per E-Mail zugeschickt werden.

Zuerst musste man sich mit dem Aufbau des ArcGIS 9 Servers und dessen Entwicklungsumgebung auseinandersetzen. Die Entwicklungsumgebung besitzt eine große Anzahl von Schnittstellen, wobei man hier schon das breite Spektrum der Funktionalitäten überblicken kann.

In der nächsten Phase musste man sich in die relevanten Schnittstellen der Programmierung einarbeiten, um das Zusammenspiel der einzelnen Objekte (Klassen, Methoden, Eigenschaften) zu verstehen. Die Firma ESRI stellt dazu Code-Beispiele bereit, die ab und zu kleinere Fehler enthalten.

Weitergehend, anhand des PageLayoutViewer Templates, wurden verschiedene Export-Funktionen getestet. Hier stellte sich heraus, dass das Impersonation Control, welches die Berechtigung des Anwenders prüft, nicht funktioniert. Daher konnte man in dieser wichtigen Testphase nicht gleich zum gewünschten Ziel gelangen.

Nach erfolgreichem Beheben dieses Fehlers wurde die Export –und E-Mail-Funktion anhand eines Kartenwerkes erfolgreich getestet.

Zum Abschluss wurden noch mögliche Erweiterungen beschreibend erklärt, welche das Auswählen des Papierformates, der Orientierung, das Auswählen des Maßstabes und die Vorgehensweise einer Legendengenerierung beinhaltet.

Anhand der Fehlerhaftigkeit einzelner Codes und Komponenten der Software, konnten leider diese Erweiterungen nicht in die Praxis umgesetzt werden.

Die Firma ESRI hat mit dem ArcGIS 9 Server eine mächtige Software entwickelt, die sich auch, wie jede neue Software, mit den üblichen „Kinderkrankheiten" auseinandersetzen muss.

Anhand der Thematik der Diplomarbeit konnte deutlich werden, dass diese Software sehr flexibel eingesetzt werden kann und der Entwicklungsspielraum noch nicht ausgeschöpft ist.

Ich denke aber, dass diese Diplomarbeit im Bereich der „Print on demand"-Anwendungen ein gutes Fundament bietet, um darauf aufbauen zu können.

8 Tabellenverzeichnis

Tabelle 1 EsriPageFormID, Seite 41

Tabelle 2 esriUnits, Seite 41

9 Literatur

1 Praxishandbuch_WebGIS, Seite 41

2 Vorlesungsskript, GIS II

5 ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch5.pdf, Seite 1

7 ms-help://ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, Seite 1

8 ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 39

10 ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 46

11 ms-help://MS.VSCC.2003/ESRI.ArcGIS/ArcGISServer/ServerDevGd_Ch2.pdf, S. 47

13 ms-help://ESRI.ArcGIS/esriDisplay/html/Display_overview.htm, DATUM: 08.03.2005

15 ms-help://ESRI.ArcGIS/esriDisplay/html/IDisplay.htm, DATUM: 08.03.2005

16 ms-help://ESRI.ArcGIS/esriDisplay/html/IScreenDisplay.htm, DATUM: 08.03.2005

17 ms-help://MS.VSCC.2003/ESRI.ArcGIS/esriDisplay/html, DATUM: 09.03.2005
IDisplayTransformation_DeviceFrame.htm, DATUM: 09.03.2005

18 ms-help://ESRI.ArcGIS/esriGeometry/html/Geometry_overview.htm,
DATUM: 09.03.2005

19 ms-help://ESRI.ArcGIS/esriGeometry/html/IEnvelope.htm, DATUM: 10.03.2005

20 ms-help://ESRI.ArcGIS/esriCarto/html/Carto_overview.htm, DATUM: 10.03.2005

21 ms-help://ESRI.ArcGIS/esriCarto/html/IActiveView_Output.htm, DATUM: 10.03.2005

22 ms-help://ESRI.ArcGIS/esriCarto/html/IPageLayout.htm, DATUM: 11.03.2005

23 ms-help://ESRI.ArcGIS/esriCarto/html/IPage.htm, DATUM: 11.03.2005

24 ms-help://ESRI.ArcGIS/esriOutput/html/IExport.htm, DATUM: 11.03.2005

25 ms-help://ESRI.ArcGIS/esriOutput/html/Output_library.htm, DATUM: 12.03.2005

26 ms-help://ESRI.ArcGIS/esriOutput/html/IExport.htm, DATUM: 17.03.2005

27 ms-help://ESRI.ArcGIS/esriServer/html/Server_overview.htm, DATUM: 18.03.2005

28 ms-help://ESRI.ArcGIS/esriServer/html/IServerContext.htm, DATUM: 19.03.2005

29 ms-help://ESRI.ArcGIS/ESRI.ArcGIS.Server.WebControls.html, DATUM: 20.03.2005

30 ms-help://ESRI.ArcGIS/ESRI.ArcGIS.Server.WebControls/
WebPageLayoutMembers.html, DATUM: 20.03.2005

10 Literatur Online

4 http://www.dotnetframework.de/default2.aspx DATUM: 05.03.2005

6 http://www.esri-germany.de/products/arcgis/index.html DATUM: 06.03.2005

9 http://www.esri-germany.de/products/arcgis/index.html DATUM: 07.03.2005

12 http://www.tks.at/technologien/objectmodel.html DATUM: 07.03.2005

14 http://www.cpp-tutor.de/mfc/mfc/kap5/lektion1.htm DATUM: 08.03.2005

31 http://msdn.microsoft.com/library/deu/cpref/html/frlrfsystemwebsessionstate.asp DATUM: 20.03.2005

32 http://msdn.microsoft.com//library/DEU/cpref/html/frlrfsystemwebmail.asp DATUM: 22.03.2005

33 http://msdn.microsoft.com/library/DEU/cpref/html/frlrfsystemwebmailsmtpmailclasstopic.asp DATUM: 23.03.2005

34 http://www.galileocomputing.de/glossar/gp/ DATUM: 24.03.2005

35 http://www.galileocomputing.de/glossar/gp/ DATUM: 26.03.2005

11 Abkürzungsverzeichnis

Abb.	Abbildung
ADF	Application Developer Framework
API	Application Programming Interface
ArcGIS	Geoinformations System der Firma ESRI
ArcIMS	Internet Mapping Server der Firma ESRI
DPI	Dots per inch
ESRI	Environmental Systems Research Institute
HTML	Hypertext Markup Language
JPEG	Joint Photographic Experts Group
LfU	Landesanstalt für Umweltschutz
PNG	Portable Network Graphic
SOC	Server Object Container
SOM	Server Object Manager
URL	Uniform Resource Locator
usw.	und so weiter
XML	Extended Markup Language
z.B.	zum Beispiel