

Programmieren von Desktop-GIS

Desktop-GIS sind und bleiben Expertenwerkzeuge, die auf die Bedürfnisse der Kunden zugeschnitten werden müssen. Zwei potenzielle Wege führen dabei zum Ziel: klassisch, durch die Verwendung von Makros, oder zweitens durch die Programmierung von Komponenten.

Im Gegensatz zu anderen Computerprogrammen aus dem Büroalltag sind professionelle GIS keine schlüsselfertigen Systeme, die man einfach installieren, starten und mit etwas Computererfahrung intuitiv bedienen kann. Aufgrund ihrer umfangreichen Funktionalität werden sie auch in Zukunft Expertensysteme bleiben, die ein entsprechendes Know-how zur fachgerechten Bedienung voraussetzen, sobald es über die einfache Datenvisualisierung hinausgeht. Für den praktischen Einsatz müssen die Systeme daher an die speziellen Nutzeranforderungen angepasst werden, sei es, um Arbeitsabläufe zu automatisieren, sei es um spezielle Funktionen hinzuzufügen, die standardmäßig nicht vorgesehen sind, aber zur Erledigung der Aufgaben notwendig sind.

Zwei Architekturen

Fast alle gängigen GIS besitzen heute eine Programmierschnittstelle, über die die erforderlichen Anpassungen und Erweiterungen möglich sind. Welche Programmiersprache dabei zum Einsatz kommt, hängt von verschiedenen Faktoren ab. Aus softwaretechnischer Sicht sind zwei Architekturen zu unterscheiden. Zum einen Desktop-GIS, hier sind heute prak-

tisch alle Systeme in das Betriebssystem Windows eingebettet und eng mit diesem verknüpft. Dies gilt etwa für Produkte wie ArcGIS, GeoMedia oder MapInfo. Anders bei Internet-GIS, also Client-Server-Systemen, die mit einem Web-Browser als Benutzeroberfläche über Internetprotokolle auf einen Server zugreifen, wie GeoMedia WebMap, ArcIMS, aber auch Open-Source-Produkte wie der UMN MapServer oder das Projekt Degree. Da im Bereich der Internet-Server auch ande-

re Betriebssysteme neben Windows eine wichtige Rolle spielen, kommt hier der Cross-Plattform-Entwicklung eine größere Bedeutung zu. Hier sind neben ASP daher auch Java und C++ im Einsatz, wohingegen im Desktop-Bereich COM (Component Object Model) beziehungsweise .NET-kompatible Sprachen Verwendung finden. In diesem Beitrag soll nur der Desktop-Bereich weiter betrachtet werden.

Makros

Zur Erweiterung und Anpassung von GIS im Desktop-Bereich sind zwei Konzepte zu unterscheiden. Klassisch ist die Verwendung von Makros, die etwa mehrere Befehle oder Abläufe in einem Skript zusammenfassen. Im engeren Sinn versteht man darunter eine Kombination einzelner Anweisungen, das heißt eine Folge von Befehlen und Vorgängen beziehungs-

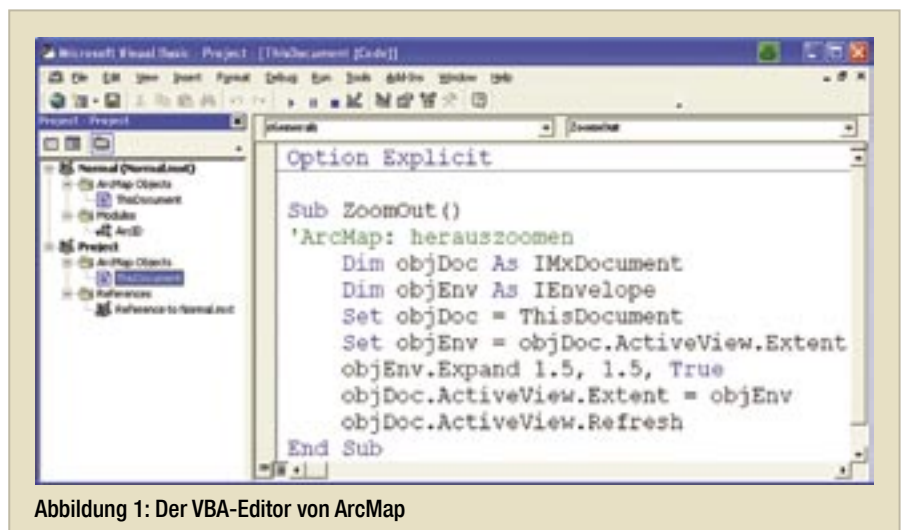


Abbildung 1: Der VBA-Editor von ArcMap

über ist es möglich, das Makro auch anderen GIS-Projekten zur Verfügung zu stellen. Mithilfe der integrierten Entwicklungsumgebung lässt sich das Makro debuggen und testen ohne die GIS-Umgebung verlassen zu müssen. Über VBA ist in der Regel auch der Zugriff auf globale Objekte des GIS-Objektmodells möglich, somit steht im Prinzip das gesamte GIS-Framework zur Verfügung. Über den Zugriff auf externe Type-Libraries lassen sich unter anderem neue zusätzliche Masken der Benutzeroberfläche mit weiteren ActiveX Komponenten gestalten.

Abbildung 1 zeigt den integrierten VB-Editor von ArcMap. Im Projektfenster links ist zu wählen, ob das VBA-Makro in der Normal-Vorlage gespeichert werden soll, und damit allen darauf beruhenden ArcMap-Projekten zur Verfügung stehen soll, oder nur im aktuellen Dokument.

Makros besitzen aber auch einige Nachteile. Da sie an eine Dokumentdatei oder eine Vorlage gebunden sind, ist ihre Verteilung nicht besonders flexibel. Außerdem liegt der Code im Textformat vor. Er kann zwar vor dem Anwender versteckt werden, steht dann aber nicht mehr für Änderungen zur Verfügung.

Komponenten

Diese Nachteile besitzt eine Erweiterung des GIS durch Programmierung von Komponenten nicht. Unter einer Komponente versteht man allgemein ein Software-Modul, das einen standardisierten Mechanismus zur Interaktion mit anderen Softwaremodulen verwendet. Die Einführung der komponentenbasierten Softwareentwicklung vor etwa zehn Jahren brachte eine erhebliche Erleichterung gegenüber der bisher üblichen Vorgehensweise in der Programmentwicklung. Bis dahin war es erforderlich, alle Module eines Programms entsprechend ihren Abhängig-



Abbildung 3: Dialogfenster des GeoMedia Assistenten für VB

keiten zu kompilieren und zu einem einzigen ausführbaren Programm zu binden. Jedes Mal wenn der Entwickler eine Änderung im Programmcode machte, musste die gesamte Anwendung erneut kompiliert und gebunden werden. Durch binärkompatible Komponenten ist es nun möglich, Komponenten unabhängig voneinander zu entwickeln. Erst zur Laufzeit wer-

den die Komponenten dynamisch hinzugeladen. Hierzu ist eine weitgehende Standardisierung der Schnittstellen zwischen den Komponenten erforderlich, sowie eine Infrastruktur, die das Auffinden der Komponenten ermöglicht.

Praxisrelevant sind in erster Linie zwei Realisierungen: JavaBeans, aufbauend auf der Programmiersprache Java, und COM/ActiveX von Microsoft beziehungsweise deren Weiterentwicklung in der .NET-Architektur. Allen diesen Ansätzen ist letztendlich gemein, Standards zu beschreiben,

die die Kommunikation von Komponenten ermöglichen. COM, bereits vor zirka zehn Jahren eingeführt, ist heute noch weitgehend der Standard im Desktop-Bereich. So beruhen zum Beispiel die aktuellen Desktop-Produkte der GIS-Marktführer Esri und Intergraph auf der COM-Technologie. Über Kompatibilitätsklassen, die .NET zur Verfügung stellt, kön-

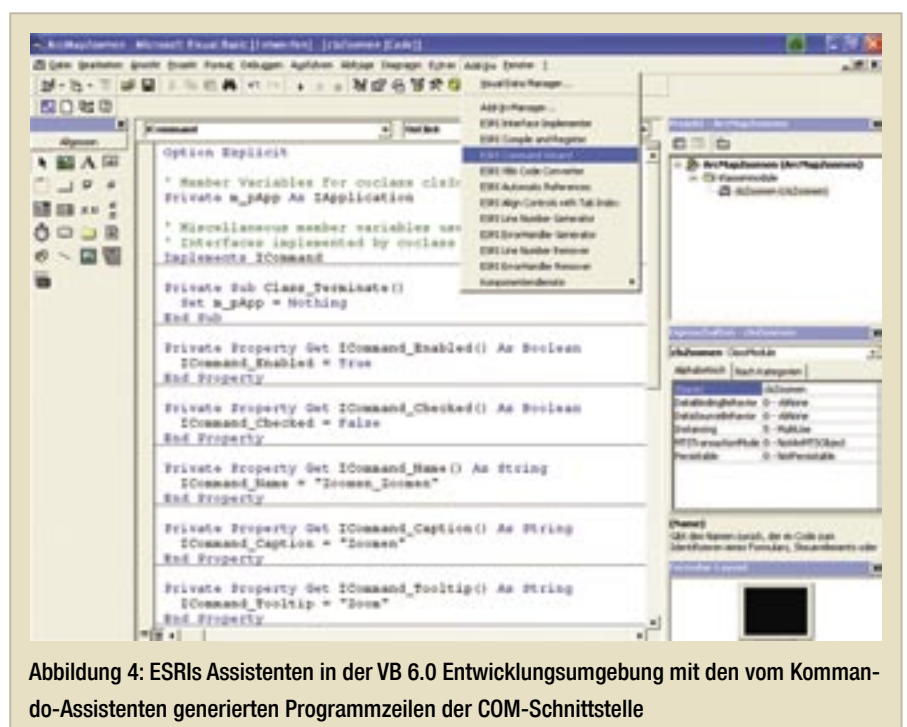


Abbildung 4: ESRIs Assistenten in der VB 6.0 Entwicklungsumgebung mit den vom Kommando-Assistenten generierten Programmzeilen der COM-Schnittstelle

nen aus .NET heraus auch weiterhin COM-Komponenten angesprochen werden. Wenn zwar auch etwas umständlich, das heißt mit zusätzlichem Overhead verbunden, ist so eine Abwärtskompatibilität gegeben, oder anderes ausgedrückt, eine COM-Schnittstelle ist auch eine .NET Schnittstelle.

Da der Ansatz von COM ist, binärkompatibel zu sein, ist eine COM-Schnittstelle nicht auf eine einzelne Programmiersprache beschränkt. Vielmehr kann die Programmierung in jeder Sprache erfolgen, die es ermöglicht COM-Komponenten anzusprechen, also etwa Visual C++, Visual Basic oder Delphi, allerdings in einer externen Entwicklungsumgebung wie zum Beispiel Visual Studio. Während in der COM-Umgebung nicht alle Programmiersprachen gleichwertig sind, da sie nicht auf dieselben Basisbibliotheken zugreifen können, entfällt dieser Unterschied in .NET. Hier bauen alle Sprachen auf derselben Laufzeit-Bibliothek auf. In der .NET-Architektur wird sogar die Anzahl der möglichen Programmiersprachen nochmals erweitert. Zu erwähnen ist hier vor allem C#, eine Eigenentwicklung von Microsoft, die vom Hersteller sehr für die .NET-Entwicklung propagiert wird.

Moderne GIS beruhen selbst auf Komponenten, das eigentliche GIS stellt sich dem Anwender als Anwendungsrahmen dar, der die verschiedenen Komponenten integriert. Der Vorteil dieses Konzepts ist es, das der Entwickler im Prinzip

auf alle Komponenten des Rahmens Zugriff hat und einerseits diese in seinen Erweiterungen und Anpassungen verwenden kann, andererseits selber Komponenten in den Rahmen einfügen kann. Der Zusammenhang zwischen den verschiedenen Komponenten wird in Objektmodellen beschrieben. Diese können sehr umfangreich sein, so dass es neben der COM-Technologie eine wesentliche Einstiegschürde für den Programmierer ist, sich einen Überblick über die Schnittstellen der Komponenten zu verschaffen. Abbildung 2 enthält einen Ausschnitt des Objektmodells von GeoMedia. Umfangreiche Objektmodelle werden heute meist mit Hilfe der Beschreibungssprache UML (Unified Modeling Language) dargestellt.

Eine gewisse Entlastung bei der Erstellung von Komponenten bieten Assistenten, die als Add-Ins direkt in die VB-Entwicklungsumgebung integriert werden können. Diese entbinden den Programmierer weitgehend davon, sich mit den Interna der COM-Infrastruktur beschäftigen zu müssen. Sie generieren bereits den zur Kommunikation zwischen der Komponente oder dem GIS-Framework notwendigen Code, das heißt sie implementieren die dafür nötige Schnittstelle. Auch die Registrierung der Komponenten wird dem Entwickler abgenommen beziehungsweise erleichtert. Der Entwickler kann sich somit ganz auf seine Aufgabe konzentrieren, der Umsetzung seines Algorithmus und dem Design der Benutzer-

schnittstelle. Gerade dem nicht so erfahrenen Programmierer sei daher die Programmiersprache VB als Einstieg empfohlen, die Auseinandersetzung mit den umfangreichen Objektmodellen ist noch anspruchsvoll genug.

Ausblick

Es ist zu erwarten, dass auch mittelfristig die Trennung zwischen Desktop-GIS und Internet-GIS bestehen bleibt. Auch wenn zum Beispiel mit ArcObjects von Esri ein einheitliches Objektmodell für beide Welten zur Verfügung steht, werden die Entwicklungsplattformen aufgrund der Kopplung an die verschiedenen Komponenteninfrastrukturen unterschiedlich bleiben. Im Desktop-Bereich wird .NET in Zukunft die COM-Technologie ablösen, wenn gleich aufgrund des recht großen Umstellungsaufwands und der im Prinzip bereits gegeben Interoperabilität zwischen .NET und COM-Komponenten viele Hersteller sich zögerlich verhalten. COM wird daher wohl noch für einige Jahre der Standard für komponentenbasierte Desktop-GIS bleiben.

AUTOR

Prof. Dr. Dietrich Schröder
Professor für Geoinformationssysteme
und Räumliche Datenbanken
Hochschule für Technik Stuttgart
Schellingstr. 24, 70174 Stuttgart
E-Mail: dietrich.schroeder@hft-stuttgart.de
Tel.: +49 (0) 711 89 26 2612