# SVG 技术在 WebGIS 中的应用研究


## 摘要：

SVG 是新兴的矢量图形，是 W3C 的推荐标准。作为矢量图形，SVG 非常适合做地图开发，在 WebGIS 方面的应用具有非常良好的前景。

本文通过对 WebGIS 的分析介绍，阐述了 WebGIS 的开发模式，技术等等，并将 SVG 的发展、特点、以及语法规则进行了深刻的分析和介绍，同时对目前业内比较流行的几种矢量图形进行了比较。

本文在客户端使用 Javascript 脚本技术对 SVG 进行研究和开发。服务器端使用 MapObject、和 Java 组件对 SVG 进行生成和分析。通过一个个试验案例证明，将 SVG 应用于 WebGIS 开发中，使 WebGIS 的功能、易用性得到极大提高，将极大促进 WebGIS 的发展。

本文最后对基于 SVG 的 WebGIS 与基于 ArcIMS 的 IMAGE 服务的 WebGIS 系统优缺点进行了比较，并认为 SVG 将对 WebGIS 的开发应用起到非常良好的促进作用。


**关键词：SVG；WebGIS；矢量图形**

# Research on Application of SVG in WebGIS

**By Wu Binzhuo**

**Directed by Xia Bin**

## Abstract

This Paper focuses on how to take advantage of SVG (Scalable Vector Graphics) to overcome the flaws of current WebGIS (Web Geographic Information System), making WebGIS serve the web map publishing (transfers, display, analysis) better.

WebGIS is an increasingly important component of GIS implementation. Nowadays, three strategies can be employed to add GIS functionality to the Web: server-side strategy, client-side strategy, server-client combination strategy. These strategies have their own advantages. In these three strategies, the server-client combination strategy integrates the advantages of the other two, but large amounts of data and applets' transferring is troublesome. It hasn't been solved until SVG is put forward. SVG is an XML grammar for describing two-dimensional graphics. It includes elements for vector shape features, raster images, animation, and text; all specified in a W3C authorized public Data Type Definition or DTD. SVG provides a new technical foundation for Internet based GIS.

The paper puts forward a new model of WebGIS which integrating with SVG. This model has abundance of client display mode, takes full advantage of the client resources to process GIS affair, and is less affected by the network bandwidth.

**Keywords: SVG，WebGIS，Vector Graphics**

# 引言

## 0.1 问题的提出

地理信息系统(Geography Information System， 简称 GIS)是 60 年代提出并逐渐发展起来的一种采集、存储、管理、分析和输出各种地理信息的信息系统。它是集中了计算机、测绘、遥感、地理学、空间科学、信息科学和管理科学等多种专业技术的交叉学科[1]。它把地理位置、地理实体与相关属性有机的结合起来，准确真实、图文并茂地将地理模拟环境展示给用户。借助其独有的空间分析能力和可视化表达，用户可以进行地理信息搜集、统计、并为决策提供支持。GIS 的这些功能，已使其成为现代社会中必不可少的基础软件之一，渗透到生产生活中的各个领域。近年来随着信息技术的迅猛发展，GIS 技术在城市建设、规划、环境资源利用、决策支持、旅游等方面发挥了重大作用。

当前网络已不仅仅是一种单纯的技术手段，它已经渗透到人们工作、生活的各个角落，并影响着人们的工作、生活方式。随着地理信息系统与各种新技术的结合，地理信息网络化的思想也得到了迅速的发展，人们希望能够在互联网上提供地理信息，让用户通过浏览器获得地理信息系统中的数据和功能服务。基于广大用户对地理信息网络化的需求，WebGIS（万维网地理信息系统）应运而生。

目前网络地理信息系统是地理信息系统领域内的热门课题之一。网络地理信息系统融合了 GIS 技术、网络技术和数据库技术，以新的工作模式和新的数据共享机制，广泛应用于各种涉及地理信息的领域，为全球范围用户提供数据、通讯和其它信息服务。

WebGIS 的实现技术多种多样。最早的 WebGIS 实现技术是服务器端实现类型，服务器端实现的 WebGIS 就是在服务器端执行 GIS 计算与处理，并把执行的结果转换为 GIF/JPEG 图像，返回客户端。目前服务器端应用包括 3 种主要技术方案：GIS 桌面系统扩展、基于 ActiveX 组件的 GIS 服务器和基于 Java 的 GIS 服务器。各种技术方案具有自己的特色，但是各种技术方案都有自己的数据格式，自己的数据处理方式，很难做到数据共享和统一。目前在 Internet 上发布 GIS 矢量地图数据大多采用的是基于 Applet 的或者是基于插件的矢量地图发布方法。这两种方法都因为是为了某一种特定格式的数据及其应用而设计的，相互之间数据不兼容，这给应用程序之间的互操作性带来了很大的麻烦。为了实现更高效、

更方便的矢量地图发布方法，人们开始将注意力转向新一代 Web 矢量图像标准 SVG（Scalable Vector Graphics，可升级矢量图形）。

SVG 是 W3C 组织为适应 Internet 的飞速发展而制定的一套基于 XML 语言的可缩放矢量图形语言描述规范。SVG 的语法和结构基于 XML，这使得它在 Web 领域具有一种先天的优势，能够满足 Web 开发者对动态、可缩放与平台无关的 Web 内容表现和交互手段日益增长的需要。同时，SVG 基于纯文本的特性具备了像文字信息一样的可检索性，从而使人们对 Web 图形进行检索。SVG 的产生，形成了一个新的网络图形标准，并且其技术标准已经为 MapInfo，ESRI，Intergraph 等 GIS 专业软件商和 Microsoft，Adobe 等其它专业软件商所接受，并且开发出了 SVGMapMaker 等基于 SVG 的地图工具和 Adobe 的 SVGViewer，IBM 的 SVGViewer 等网络浏览插件。

**0.2 文章的组织**

本文由以下几分组成：

引言：简述了 GIS 的发展以及 WebGIS 的产生的重要意义以及 WebGIS 目前存在的一些问题。简单叙述了 SVG 的产生背景以及特点，指出其在 WebGIS 领域广阔的应用前景。

第一章：通过介绍 WebGIS 的产生与发展，比较了常用的 WebGIS 开发模式和技术。

第二章：介绍了 SVG 技术的产生与发展，以及其特点。

第三章：使用 SVG 建立 GIS 数据模型，并做地理数据表现试验。

第四章：结合 MapObject 和 SVG，建立 WebGIS 系统原型，并就其中的客户端技术细节进行探讨。

第五章：总结全文，并对 SVG 在 WebGIS 中的应用前景做出展望。

# 第一章 WebGIS 的产生与发展

GIS 技术和 Internet 技术的融合，使产品的应用环境、产品的概念、产品软件的结构都发生了很大变化。与传统的 GIS 相比，WebGIS 具有访问范围更广泛、实时性更强、数据可分布管理、操作更简单、能适应于不同的软硬件平台、降低系统成本等优势。

## 1.1 WebGIS 概述

WebGIS 是 Internet 技术应用于 GIS 开发的产物，是实现空间数据互操作的一条极佳解决途径。从 Internet 的任意一个节点，用户都可以浏览 WebGIS 站点管理的空间数据，进行制作专题图等各种空间信息检索和空间分析。WebGIS 不但具有大部分乃至全部桌面 GIS 软件具有的功能，而且还具有利用 Internet 优势，使得用户不必在自己的本地计算机上安装 GIS 软件就可以通过 Internet 访问远程的 GIS 数据和应用程序，进行 GIS 分析，在 Internet 上提供交互的地图和数据[2][3]。

WebGIS 的关键特征是面向对象、分布式和互操作。任何 G1S 数据和功能都是一个对象，这些对象部署在 Internet 上的不同服务器上，当需要时进行连接和调用。Internet 上的任何其他系统都能和这些对象进行交换和交互操作。

### 1.1.1 WebGIS 的基本特征

- **WebGIS 是集成的全球化的客户/服务器网络系统**

WebGIS 应用客户/服务器模式来执行 GIS 的分析任务。它把任务分为服务器端和客户端两部分。客户通过客户端向服务器请求数据、分析工具或模块，服务器或者执行客户的请求并把结果通过网络送回给客户，或者把数据和分析工具发送给客户端供客户使用。

- **WebGIS 是交互性**

单纯的 Web 服务，用户只能通过超链接浏览由 WWW 发布者发布的静态图形和文本，这些图形大部分是 JPEG 和 GIF 格式的文件，用户无法操作地图，甚至连像 Zoom、Pan、Query 这样简单的分析功能都无法执行。但 Web 服务提供了用户与网络的交互性。

WebGIS 使用户可以在 Internet 终端操作 GIS 地图和数据，用 Web 浏览器(IE、Netscape，等等)执行 WebGIS 服务器提供的 GIS 功能：如 Zoom（缩放）、Pan（拖动）、Query（查询）和 Label（标注），甚至可以执行空间查询：如“离你最近的旅馆或饭店在哪儿”，或者更先进的空间分析：比如缓冲分析和网络分析等。在 Web 上使用 WebGIS 就和在本地计算机上使用桌面 GIS 软件一样。

- **WebGIS 具有分布式性**

GIS 数据和分析工具是独立的组件和模块，WebGIS 利用 Internet 的这种分布式系统把 GIS 数据和分析工具部署在网络不同的计算机上，用户可以从网络的任何地方访问这些数据和应用程序。在网络世界里，要求每个站点的用户在自己的计算机里拥有所有的地球信息资源无疑是一种极大的浪费，同时也必然导致频繁的系统维护造成的人力浪费。WebGIS 应该能够使得同一 GIS 系统的用户通过网络不但能够访问本系统中分散在不同站点的数据和数据处理服务，同时还能维持系统的一致性以及分散大量访问带来的网络负载。

- **WebGIS 具有可更新性**

由于 WebGIS 是分布式系统，数据库和应用程序部署在网络的不同计算机上。地理数据和地理信息服务可以由管理员进行更新维护。用户访问到的都将是最新可用的数据和功能，即只要数据源或者功能组件发生变化，WebGIS 将得到更新。WebGIS 和数据源的动态链接，WebGIS 组件的重新部署将保持数据和软件的现势性。

- **WebGIS 是跨平台系统**

WebGIS 可以被拥有任何硬件机器，任何操作系统的用户所访问。只要能访问网络，用户就可以访问和使用 WebGIS 而不必关心用户运行的操作系统或者硬件设备什么。随着 J2EE 技术的发展，未来的 WebGIS 可以做到“一次编写，到处运行”，使 WebGIS 服务器能够部署在任何软硬件系统下，这将使 WebGIS 的跨平台性提高到一个新的层次。

WebGIS 不仅能够跨硬件平台、操作系统平台，未来的 WebGIS 还能够在异构环境下访问和共享 GIS 数据、功能和应用程序。这需要地理数据具有很高的互操作性。OGC 提出的开放式地理数据互操作规范（OpenGeodata Interoperability Specification）为 GIS 互操作性提出了基本的规则。其中有很多问题需要解决，例

如数据格式的标准、数据交换和访问的标准、OIS 分析组件的标准规范等。随着 Internet 技术和标准的飞速发展，能够跨数据访问的 WebGIS 将会成为现实。

- **WebGIS 是图形化的超媒体信息系统**

使用 Web 上超媒体系统技术，WebGIS 通过超媒体热链接可以链接不同的地图页面。例如，用户可以在浏览全国地图时，通过单击地图上的热链接，而进入相应的省地图进行浏览。

另外，WWW 为 WebGIS 提供了集成多媒体信息的能力，把视频、音频、地图、文本等集中到相同的 Web 页面，极大地丰富了 GIS 的内容和表现能力。

### 1.1.2 WebGIS 的基本要求

- **WebGIS 应当是开放的**

WebGIS 能够共享多种来源、多级尺度（比例尺）、存放在不同地点的地理数据，能够和其他应用软件集成，并通过 Java、CORBA、DCOM 等技术跨平台协作运行，支持 B/S 模式等。

- **WebGIS 能在 Internet 环境下运行**

WebGIS 使用 Internet 协议标准，将 GIS 服务器与 Web 服务器集成，通过普通浏览器，用户可以在任何节点访问 WebGIS，共享地理空间信息服务，从而将 GIS 扩展成为公众服务系统。

- **WebGIS 必须支持数据分布和计算分布**

WebGIS 服务器为网络用户提供 GIS 服务：地理数据存取服务、地理数据目录服务、地理信息分析服务和地图显示服务。通过互操作技术，共享分布的数据对象，在多个不同的平台上协同运行，最大限度地利用网络资源。

- **WebGIS 能在网络上直接查询和存取数据**

建立地理时空数据结构标准和操作标准，直接在 Internet 上查询数据和存取数据。

### 1.1.3 WebGIS 的基础技术

- **空间数据库管理技术**

对象-关系数据库技术和面向对象的数据库技术逐渐成为数据库技术的主流，

也成为管理 GIS 空间数据的主要技术。关系型数据库管理系统已经相当成熟。商业化的关系型数据库支持数据分布，通过 SQL 语言和数据库驱动，所有的 GIS 组件都能连接数据库，使用数据库中的空间数据。目前比较强大的空间数据库 Oracle Spatial 是其中的代表。

- **面向对象方法**

地理实体、空间实体非常适合用面向对象的思想来描述。它集抽象性、封装性、继承性和多态性于一体，可以帮助人们开发出模块化、数据抽象程度高的，体现信息隐蔽、可重用、易修改、易扩充等特性的程序。面向对象是一种模拟方法。面向对象分析(OOA)、面向对象设计(OOD)、面向对象语言(OOL)和面向对象数据管理(OODBM)贯穿整个信息系统的生命周期。面向对象的空间数据库技术正在逐步成熟，空间对象查询语言(SOQL)、空间对象关系分析、面向对象数据库管理、对象化软件技术等，都和 GIS 密切相关。

- **浏览器/服务器模式**

在 C/S 时代，多种网络平台让用户平台之间的互相连接变得异常复杂。C/S 不是基于网络的技术，因此发展出众多不兼容的技术平台。每种技术平台都只是针对具体的应用环境和需求，为自身的维护和发展制造了巨大的苦难。进入网络计算机时代后，互联网技术使网络获得了高度稳定性。

B/S 模式的实现技术本质上是一种客户机技术，这对于大中型企业特别合适。在 C/S 模式下，应用的主要部分是在客户端。千差万别的客户端，让客户端管理变得复杂多变，工作量很大。而 Internet 技术却不同，用户都通过浏览器这个统一平台，共享同样的永远在不断发展变化着的信息服务。Intranet 从某种意义上上取消了所有在客户机端的维护工作。

- **组件技术**

组件技术的基本思想是：创建和利用可重用的软件组件来解决应用软件的开发问题。组件是一种可复用的小软件（可以是二进制的，但不是源代码）。组件可以有多种多类，小到图形界面上的一个按钮，大到一个复杂的软件。众多的组件组成一个应用程序。

组件的出现提高了开发速度，降低了开发成本，增加了应用软件的灵活性，降低了维护费用。

- 分布式计算机平台

即 Distributed Computing Platform 技术，目前有 OMG 的 CORBA/Java 标准和微软的 DCOM/ActiveX 标准。

- 其他技术

多媒体数据操作标准 ISO SQL/MM、地理数据目录服务技术（Geodata Catalog Service）、数据仓库技术、地理信息高速公路技术等

## 1.2 WEBGIS 基本框架

WebGIS 的基本框架如图 1 所示：

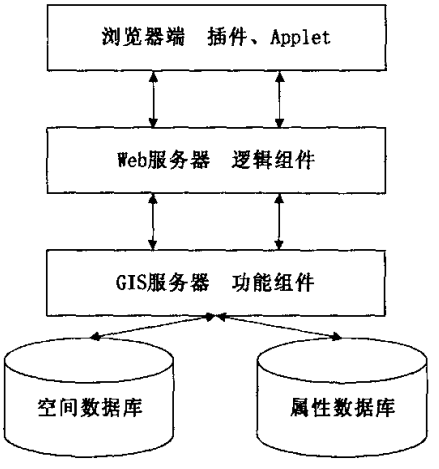


图 1. WebGIS 基本框架

WebGIS 的客户端是网页浏览器。通过安装 GIS Plug-In、下载 GIS ActiveX 或 GIS Java Applet，实现客户端的 GIS 计算，或者直接接受服务器发回的图片和文本信息，展示地理信息。WebGIS 服务器端由 Web 服务器、GIS 服务器、GIS 数据库服务器组成。其中：

Web 服务器接受来自浏览器的请求，传递给 Web 服务器，Web 服务器调用 GIS 服务器的组件或者其它组件，完成用户的请求。GIS 服务器以及其它组件连接空间数据库和属性数据库，根据用户的需要去请求数据服务器，然后将结果发送回服务器，这些结果以文本、图片等多媒体格式由服务器返回给浏览器。GIS 服务器也能同客户端的 GIS Plug-In/ActiveX/Java Applet 直接通信，完成 GIS 服务。

实质上，WebGIS 中有两大要素：GIS 数据和 GIS 计算。实现 WebGIS，就是

如何合理地在服务器或者浏览器端的插件上进行 GIS 数据操作和 GIS 计算。GIS 计算可以分割为服务器端部分和客户端部分，浏览器端部分的 GIS 计算通过网络，从服务器迁移到客户机执行。对 GIS 计算的策略不同，WebGIS 实现的技术方案也就不同。GIS 数据也可以下载到浏览器端进行显示。这些实现技术大致分为客户端实现和服务器端实现，或者混合实现。

## 1.3 WEBGIS 的客户、服务器实现技术

### 1.3.1 WebGIS 的客户端实现技术

客户端 WebGIS 允许用户在本地计算机的浏览器上进行 GIS 的数据和 GIS 计算。GIS 数据和 GIS 数据操作组件开始存储在服务器上，用户通过 Http 协议从服务器请求这些数据和组件，接到服务请求后，服务器把数据和组件传给用户，由用户在本机进行 GIS 数据的显示和操作。客户端应用包括 3 种主要技术方案：插件程序、ActiveX 控件和 Java Applet。

- 插件程序

插件工作在 Netscape 和 IE 这样的 Web 浏览器上用于处理 GIS 数据和地图的程序。它们从网络服务器下载并安装到用户的本地计算机上，在浏览器中运行。GIS 插件处理嵌在 HIML 中的 G1S 数据(以特殊的形式存在：如 XML)，而这些特殊的 G1S 数据浏览器本身不能识别。

GIS 插件的作用是为浏览器提供新的无缝的 GIS 数据支持，能够使 Web 浏览器能够显示 GIS 数据，更方便的处理 GIS 数据。

- **ActiveX 控件**

Microsoft 的 ActiveX 技术创建的 GIS 控件。这些 GIS 控件和 GIS Plug-Ins 一样，用于扩展 HTML 的功能，以便使浏览器能处理 GIS 地图和数据。

Microsoft ActiveX 文档是一种特殊的 Helper 程序。这些文档由 Microsoft Office 应用或其他的兼容应用所创建，并且可以嵌在 Web 浏览器中显示。当一个 URL 指向这样一个文档而且服务器为其配置正确的 MIME 类型时，支持这种类型文档的应用程序能直接在浏览器中自动启动运行。如：Web 浏览器碰到像 ArcView 的.shp 文件这样的文件格式时，此文件的本地程序(ArcView)就能在浏览器的窗口中启动运行，并把浏览器的菜单及工具条和自己的组合在一起。

➢ **Java Applet**

最新的开发是用 Java 编程语言来创建 Java Applet。GIS Java Applet 从服务器下载到本地计算机并在浏览器内立刻执行。用户的计算机上应安装 Java 虚拟机来解释 Java Applet。GIS 的功能和数据被封装在一个包内，与 Applet 程序一起迁移到客户端。Applet 不需要安装就可以运行。单独的 Applet 还可以和服务器进行通讯，显示、处理服务器上的地理数据，而不用将地理数据下载到客户端。

Java 是面向对象的编程语言，它更适合网络环境；Java 是跨平台的，因此 Java Applet 可以运行在绝大多数的软硬件平台下； Java2D 和 Java3D 对创建和显示图形提供了最广泛的支持，非常适合在浏览器端进行地理数据显示。

### 1.3.2 WebGIS 服务器端实现技术

服务器端应用的 WebGIS 就是在服务器端执行 GIS 数据操作，并把执行的结果转换为 HTML 格式(一般是 GIF/JPEG 图像)返回客户端。GIS 数据和 GIS 服务组件都部署在服务器上，对浏览器端请求的响应只是在服务器端进行 GIS 计算，然后将结果转化成图片和文本格式，返回给用户，而浏览器不用对这些结果做任何处理。传统的 WebGIS 实现方式有：CGI(Common Gateway Interface)、Server API、Plug-in，目前 Java 已经成为构建 WebGIS 的主流技术。

➢ **CGI 实现方式**

CGI 根据用户请求，连接服务器端 GIS 软件，GIS 软件按照要求生成一副图片，传回客户端。但是，大多数 GIS 不能直接作为 CGI 程序直接连到 Web 上，给设计开发带来了麻烦，同时这种开发技术存在着严重的扩展性问题——每一个新的 CGI 程序要求在服务器上新增一个进程。如果多个用户并发地访问该程序，这些进程将耗尽该 Web 服务器所有的可用资源，直至其崩溃。

➢ **Server-API 实现方式**

Sever API 与 CGI 有些类似， Server API 启动后一直会处于运行状态，耗用资源少，因此服务响应比 CGI 快。但是 Server API 依附于特定的服务器，可移植性很差。

➢ **Plug-in 实现方式**

Plug-in 实现方式是在浏览器端安装相应的插件用来显示从服务器端传送的矢量或栅格形式的 GIS 数据。这种方式把一部分服务器上的功能移到浏览器上，大大加快了客户操作的反应速度，而且也减少了交互网上的流量和服务器的负载。但在客户端要先安装相应的插件或控件，并且下载插件和地理数据也要耗费一定时间。

- **基于 Java 的 GIS 服务器**

目前以 J2EE 为首的多层结构已经成为主流，多层结构系统具有良好的可拓展性和可维护性，带来的是稳定的系统质量，同时，可以达到软件重用，保护投资，节省新项目的开发时间。另外 Java 的跨平台性使得基于 Java 的 GIS 服务器可以在不同的软硬件平台上移植，免去了繁复的移植工作。因此，采用 J2EE 技术开发 WebGIS 是一个很好的选择。

# 第二章 SVG 技术及其应用

SVG(Scalable Vector Graphics，可升级矢量图像)是一种基于 XML 标准的开放的矢量图形描述语言。SVG 图像是与 XML1.0 兼容的文档，SVG 元素是指示如何绘制图像的一些指令，SVG 浏览器解释这些指令，把 SVG 图像在指定设备上显示出来。安装了插件的网页上可以显示出各种各样的高质量的 SVG 矢量图形，支持非常的功能：绝大多数几何图形、动画、渐变色、滤镜效果等。最关键的是，它是完全用基于 XML 标准的格式化文本来描述的。也就是说，SVG 是一种专门为网络而设计的基于文本的图像标准。

1999 年 2 月，SVG 草案出台，经过不断地修改更新，最终形成了第一份实验性的实现规范。2001 年 7 月，W3C 正式发布了 SVG 图像格式建议书——SVG1.0 规范。目前，该规范的版本是 1.2，已经成为网络图形的推荐标准。 W3C 对 SVG 的解释是："SVG 是一种使用 XML 来描述二维图像的语言。它允许图形对象以 3 种形式存在，分别是矢量图形、点阵图像和文本。各种图像对象能够组合、变换，并且能改变其样式，也能够定义成预处理对象。文本是 XML 命名空间中的有效字符，这些字符能作为 SVG 图像的关键字在搜索引擎中搜索到。SVG 的功能包括嵌套变换、路径剪裁、透明度处理、滤镜效果以及其他扩展，同时，SVG 支持动画和交互，也支持完整的 XML 的 DOM 接口。任何一种 SVG 图像元素都能使用脚本来处理类似于鼠标单击、双击以及键盘输入等事件。因为同 Web 标准兼容的缘故，SVG 还能够在同一个 Web 页面里凭着继承自 XML 的命名空间等特性来完成一系列交互操作。"

## 2.1 SVG 的优点

任意缩放：用户可以任意地缩放 SVG 格式的图形而不会破坏清晰度；

文本独立：SVG 图形中的文字独立于图形，可以编辑、复制、查询。用户系统即使没有安装某一字体，同样可以看到这些字体；

较小文件尺寸：一般而言，SVG 图形的尺寸要比其他图像格式如 GIF，JPEG 更小；

超强色彩控制：SVG 图像具有 1600 万种颜色，支持 ICC 标准，RGB，线性填充和遮罩；

完全支持 DOM（文档对象模型）：SVG 以及 SVG 中的对象（元素）可以通过脚本语言控制，例如鼠标动作，实现自身缩放、颜色改变，样式改变等操作；

动态生成：SVG 图形文件可以用任何编程语言来生成；

基于 XML 标准：XML 是公认的下一代网络标记语言，拥有无穷的生命力。SVG 基于 XML 标准，并且同 HTML、CSS、DOM、XSL、JavaScript、CGI 一样，将成为新的标准；

灵活易用的文件格式：SVG 主要由 3 个图形对象组成：矢量图形、位图和文字。由于 SVG 文件是以文本的形式存放的，更改起来是非常方便的。也就是说，可以不用任何图像处理工具，仅仅用记事本就可以生成一个 SVG 图像。

目前浏览器还不能支持 SVG 图像的显示。为了能在普通的 Web 浏览器中观看 SVG 图形，必须安装 SVG 浏览器插件。由于 SVG 的广泛流行，支持 SVG 的浏览器、插件越来越多。

### 2.2 SVG 的应用价值

如同 PNG 作为 W3C 的位图图像工业标准，SVG 是网络中解决矢量图像的工业标准。在 SVG 出现以前，在网络中应用的矢量文件格式只有 Macromedia 公司开发的 SWF 格式。因为位图文件受到本身的很多局限，在图形印刷和传输中，矢量文件有很大的优势，因此，SVG 作为相应的矢量标准，必将在开发和应用中得到广泛使用。

作为 W3C 的推荐标准，SVG 与现有的其他开放标准有很好的兼容性。这些标准包括 DOM， CSS，XML，XPointer，XSLT，XSL，SMIL，HTML，XHTML 等。SVG 与其他组织的标准化技术也能很好的协调一致，如 ICC，URI，UNICODE，sRGB，ECMAScript/javascript，Java。使用 SVG 并不意味着我们将从此放弃现在的网络图像技术如 GIF，JPEG，SWF。相比较这些目前有普遍应用的文件格式，SVG 更适合网络发展的需求，开发和应用 SVG 意味着获得一个更优秀的工具和方法[17]。

- **数据表格，图像地图**

在应用领域，SVG 可以非常方便的应用在数据表格和图像地图中。通过文档对象模型，任何变成语言或者文本修改工具都可以很方便的控制、生成需要的图形。这为网络图像数据表格提供了很大的应用前景。一个简单的例子，现有的动

态网页可以方便的生成动态数据网页，而 SVG 可以同样方便的根据动态数据绘制图像，例如数据分析中的柱状图和饼状图。

制作地图同样是 SVG 的另外一大优势。由于 SVG 是矢量格式，图像可以在任何显示分辨率下获得同样的图像效果，图像如何放大也不会有任何损失，因而可以完全可以制作一个包含了城市所有的地理信息的地图。用户根据需要，对地图的不同地区进行放大显示，同时，每一个地理实体又可以包含一段文字说明，或者包含相应的属性数据，当用户需要时，可以通过鼠标操作获得文字说明或者其它属性信息。这些功能不需要同服务器或者数据库有任何交互动作。

- **无线设备的需求**

SVG 另一个非常诱人的应用前景就是开发无线设备的图形和动画。例如目前使用的手机产品，其图像主要是 WBMP。这种格式因为是位图文件，受到传输大小的限制，同时不提供彩色的色彩模式，所以显得很呆板。而如果利用 SVG，只要在无线设备中安装一个文本解析器，就可以实现对 SVG 文件的识别和显示，同时，因为是矢量的文本文件，文件的尺寸不会很大，非常适合无线产品的网络传输。SVG 还提供动画和多媒体编辑功能，所以可以支持二维的平面动画，支持声音文件和视频文件的播放。

- **图像搜索引擎**

不同于现在的二维图像，SVG 是一个可实现交互和查询的文件格式。在 SVG 图像中，信息是一元代码形式的，是开放形式的。文字独立于图形信息。这就为图像搜索和查询提供了可能。例如，在一个 SVG 动画中，通过搜索某一个关键字，就可以在图像中查询到对应的信息。而这对于同样是矢量图像和动画格式的 SWF 就不可能实现。依据 SVG 的这种交互性，可以创建大型的图像搜索引擎。

- **网页设计思想的改变**

采用 SVG 进行网页设计，是下一代的网页设计思想。现在的网页设计，通常是在位图图像软件中绘制好整体页面图像，然后进行图像切割，最后完成页面的文字编辑。通常这样的工作是很繁琐的，进行二次修改也很不简便。而利用 SVG，则可以实现页面图形设计和文字编辑的一步完成。如果要进行远程协作完成网页设计，也只需要传输创作后的页面文件，而不必将所有的图像源文件和页面文件都传输。

因为 SVG 能够很好的与 HTML 和 XML 兼容，所以，下一代的网页编辑软件将开始结合图像创作功能。许多现在需要通过外部图像或动画软件创作的效果，将可以直接在网页编辑软件中完成。

结合 SVG 创作网页，将使网页设计师真正可以"画"出页面。

### 2.3 SVG 的基本格式

与 HTML 相类似，基于 XML 的 SVG，语法和格式也是结构化的。所谓结构化，也就是文件中的对象通过特定的元素标签定义，任何元素都可以作为对象进行管理，文件是框架式的。掌握基本的文件框架，就可以阅读、编辑和创作自己的文件。

SVG 使用一组组的元素标签，创建和组织文件以及文件中的对象。每一个 SVG 文件都包含最外层的<svg></svg>标签。该标签用于声明 SVG 文件的开始和结束。这等同于 HTML 文件中的 HTML 文件声明标签<html></html>。下面的代码是一个通常的独立 SVG 文件的头部标识：

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20000303 Stylable//EN"
"http://www.w3.org/TR/2000/03/WD-SVG-20000303/DTD/svg-20000303-stylable.
dtd";>
<svg xml:space="preserve" width="5.5in" height=".5in">
.....
</svg>
```

其中，<?xml version="1.0" encoding="iso-8859-1"?>建立了"本文件基于 XML 1.0，编码方式是 iso-8859-1"的基本信息。

<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20000303 Stylable//EN" "http://www.w3.org/TR/2000/03/WD-SVG-20000303/DTD/svg-20000303-stylable.dtd";>虽然很长，但只要理解，它是一个注释性的提示信息。它声明了文件定义类型(DTD)。文件定义类型(DTD)是一套用于定义元素、元素属性和制定如何形成文件的标准规则。根据特定的需要，DTD 可以进行自定义。该行省略不影响 SVG 文件的生成。

```
<svg xml:space="preserve" width="5.5in" height=".5in">
...
```

</svg>用于声明 SVG 文件主体的开始和结束，同时也声明了图像容器的大小。图像容器也就是最后图形或动画的生成尺寸，相当于绘画中使用的画布。

SVG 主体中的语法和格式定义，遵循 XML 标准。通过各类元素标签和标签对应的属性来描述 SVG 对象。SVG 对象主要包括图形，路径和文本。

下面是一个调用 mp3 音乐的例子：

```
<?xml version="1.0" standalone="no"?>
        <svg xmlns:a="http : //www.adobe.com/svg10-extensions";
    a:timeline="independent"  width="100%"  height="100%"
    viewBox="0 0 620 420">
    <a:audio xlink : href="follow.mp3" volume="10" begin="0s"
repeatDur="indefinite">
    </a:audio>
    </svg>
```

将上面的源代码粘贴到 windows 的记事本，选择文件类型为所有文件，保存文件为*.svg(后缀名自己填写)，同时在文件保存的文件夹中放置一个名为 follow 的 mp3 文件，如果您的浏览器安装有 SVG 播放器，就可以听见 mp3 音乐。

### 2.4 SVG 与 Flash 的比较

Flash 是目前最流行的网络矢量图形之一。Flash 制作的动画、图片非常精美，赢得了众多用户的青睐。SVG 同样可以制作出精美的的动画和图片。Flash 最终生成二进制文件，生成以后修改十分困难，而 SVG 是基于字节码的，直接用文本工具就可打开进行编辑。相同效果的 Flash 文件一般比 SVG 文件要大。SVG 文件可以存储比 Flash 更多的信息。SVG 可以广泛的应用在手机等移动终端上，而 Flash 在其他领域的应用则很少。SVG 的文档对象模型可以很方便的被 Javascript 等客户端脚本控制，而 Flash 对脚本的支持非常有限。

当然矢量图形还有很多种，比如 VML，PDF 等等。VML 是微软开发的内嵌在 word 里的矢量图形，格式上和 SVG 比较类似，但是复杂的 VML 执行效率低下，功能上也很薄弱，而且微软已经不为它做进一步的扩展，其他浏览器平台也不支持这种矢量图形，因此起发展前景非常有限。PDF 也是非常强劲的矢量图形，由于其只是作为文档来使用，在矢量图形上的功能不是很突出。

# 第三章 基于 SVG 技术的数据模型及表现

## 3.1 SVG 的基本形状

SVG 定义了六种基本形状，每个基本形状对象都定义了位置和颜色属性，其颜色和轮廓由 fill 和 stroke 等属性决定。

这六种基本形状分别是：

- **圆形(circle)**

一个原点在指定点，半径为制定长度的标准圆形。如图 1 所示：

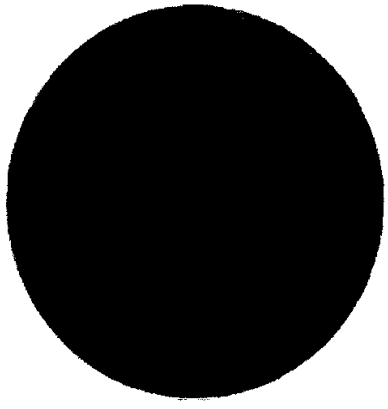

图 1. 圆形

圆形代码：

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
"http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <circle stroke="red" fill="blue" cx="50" cy="50" r="20"/>
</svg>
```

- **椭圆(ellipse)**

定义了中心点和长短轴长度的椭圆。如图 2 所示：

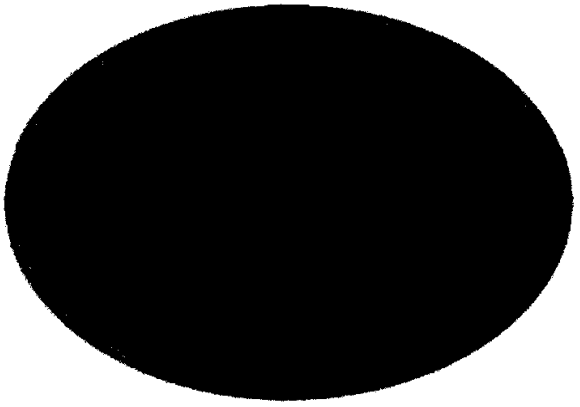

图 2. 椭圆

椭圆代码：

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
"http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <ellipse stroke="red" fill="blue" cx="50" cy="50" rx="30" ry="20"/>
</svg>
```

- 矩形(rect)

指定了左上角坐标点，宽度和高度的矩形。如图 3 所示：

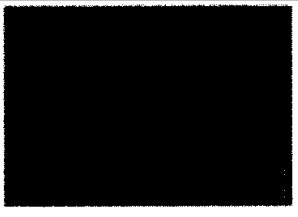

图 3. 矩形

矩形代码：

```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <rect stroke="red" fill="blue" x="50" y="50" width="30" height="20"/>
</svg>
```

- 线(line)

指定了起始点和终止点的直线。如图 4 所示：

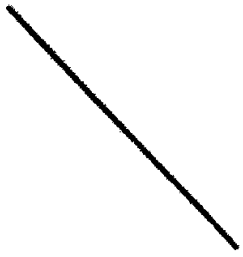

图 4. 线

线代码：

```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <line stroke="red" fill="blue" x1="50" y1="50" x2="100" y2="100"/>
</svg>
```

- 折线(polyline)

由一系列指定点连接而成的折线。如图 5 所示：

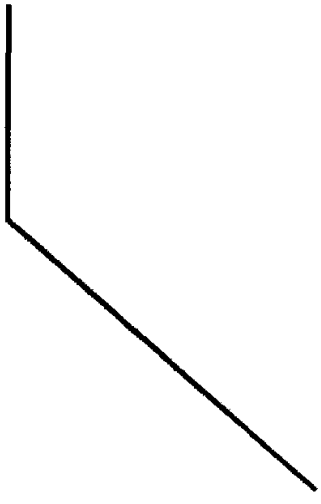

图 5. 折线

折线代码：

```
<svg viewBox="0 0 100 100" xmlns="http://www.w3.org/2000/svg">
    <polyline stroke="red" fill="none" points="10,10 10,50 70,100"/>
</svg>
```

- 多边形(polygon)

由指定的一系列点相连构成的闭合图形。如图 6 所示：

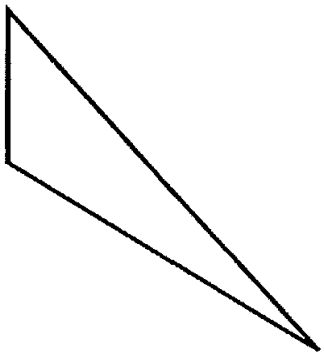

图 6. 多边形

除了以上六种基本形状以外，路径(path)也是一个很重要的形状元素。路径由一系列的点规定，该形状可以是开放的（如线）或闭合的（如多边形），并可以包含一条或多条线、曲线和线段。如图 7 所示：

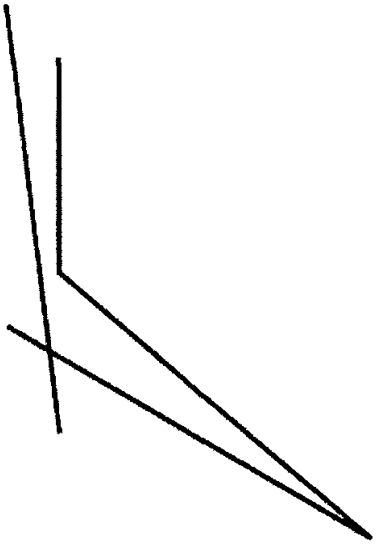

图 7. 路径

### 3.2 基于 SVG 的地理数据模型

SVG 规定的 6 种基本形状和路径互相组合，就可以画出绝大部分的地理图形。在地理要素中，主要包含的集合形状是点线面。通过 path 可以完全把线和面实现，点元素可以用 circle 图形来表示。

SVG 的图形元素支持自定义属性，因此任何地理元素的属性都可以加入到图形元素中去，例如长度，面积，日期等等。

➢ **面模型**

由于面可能包含湖或者岛，而普通的 polygon 不能描述这些特殊的地理元素。Path 是描述面的最佳模型。

面模型应包含以下数据(如表格 1 所示)：

| 名称 |
| --- |
| 坐标序列 |
| 周长 |
| 面积 |
| 颜色 |

**表格 1. 面模型**

面模型实例(图 8)：

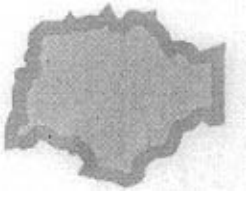

**图 8. 面模型实例**

代码:

```
<path id="名称" stroke="颜色" fill="颜色" length="周长" area="面积" d="坐标序列"/>
```

➢ **线模型**

一般的线用 polyline 来表示即可，但不排除有断断续续的线，因此用 path 来统一线的表达方式。

线模型应包含以下数据(如表格 2 所示)：

| 名称 |
| --- |
| 坐标序列 |
| 长度 |
| 颜色 |

**表格 2. 线模型**

线模型实例(图 9)：

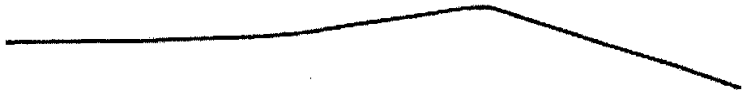

图 9. 线模型实例

代码：

```
<path id="名称" stroke="颜色" fill="none" length="长度" d="坐标序列"/>
```

- 点模型

点是没有面积的，因此其数据模型非常简单(如表格 3 所示)。在地图上，点可以有各种各样的形状，因此我们可以用 rect，circle 等等来表示。

| 名称 |
| --- |
| 坐标序列 |
| 半径 |
| 颜色 |

表格 3.点模型

点模型实例(图 10)：

代码：

```
<circle id="名称" stroke="颜色" fill="颜色" cx="坐标序列" cy=" 坐标序列" r="半径"/>
```

普通的 2D 地理元素都可以通过 SVG 来表现。SVG 还可以存储简单的地理属性，例如面积，人口，长度等等。

目前的数据大多以文件(各种公司格式不同)形式存储，或者存放在空间数据库中。通过解析，可以将其他形式的地图文件转换为 SVG 格式的图形文件。下面是成功转换后的深圳市地图，包括行政区划图和主要道路。

图 10. 深圳市 SVG 地图

深圳市 SVG 地图格式的文件大体结构如下：

```
<svg viewBox="....">
<g transform ="matrix(1 0 0 -1 0 45.268328)">
<g .....>//[illegible] 1
<path.....>
.....
</g>
<g.....>//[illegible] 2
<path. ....>
.....
</g>
</g>
```

viewBox 是控制地图显示范围的属性，viewBox 可以为地图重新设置坐标系统，即地图在浏览器上的坐标系统可以不同于屏幕坐标系统。transform ="matrix(1 0 0 -1 0 45.268328)"是非常重要的一个属性。北半球的经纬度坐标的原点是在左下方，而屏幕坐标系统是在右上方，因此必须通过矩阵变换将地图翻转。

# 第四章 WebGIS 系统设计与实验

WebGIS 的设计模式从 CGI 到组件式，从胖客户端到瘦客户端，多种多样。每种都有自己的利弊。胖客户端消耗了大量的客户机的资源，并且数据下载占用带宽，浪费了用户的时间，同时客户端使用的数据格式、标准各有不同，影响了数据共享，难以移植复用。瘦客户端功能比较弱，客户需要等待服务器的相应，影响 WebGIS 的使用效率，同时也给服务器造成了沉重的负担。

基于 SVG 的 WebGIS 系统，将数据以文本的形式下载到客户端，占用网络流量很小，大量的复杂操作可以在客户端完成，节省了网络资源、减轻了服务器负担。在 SVG 形成工业标准的今天，SVG 数据必然能够被用户广泛采用和复用，这将消除不同地理数据格式之间的差异。

## 4.1 WebGIS 系统功能结构

**1、缩放**

在客户端实现对地图放大，缩小操作，方便快捷，并且保证图像不失真。

**2、漫游**

在客户端实现对地图视图范围的移动无缝操作，方便灵活。

**3、地理实体信息显示**

鼠标漫游到地理实体上面时，将在信息框显示该地理实体基本属性。

**4、地理实体查询**

在客户端通过地理实体的名称、地理实体属性(面积、长度等等)进行查询，方便灵活，响应迅速。

**5、地理实体编辑**

在客户端操作地理实体，可以将地理实体(点、线、面)进行删除、移动、复制，甚至对地理实体的形状进行处理。操作结果可以通过服务器端的功能组件对空间数据库进行更新。对地理实体的编辑操作可视性强，接近桌面 GIS 同等程度效果，克服了以往 WebGIS 难以在客户端实现的编辑功能困难。该特点充分发挥了 SVG 作为网络图形的优点，

**6、空间拓扑查询**

空间拓扑查询功能有服务器端的功能组件完成。目前 ESRI 功能已经有非常

完善的在服务器端操作空间数据库的组件，利用这些组件，可以快速的建立起一个空间数据库操作平台。

本系统功能结构图如下：

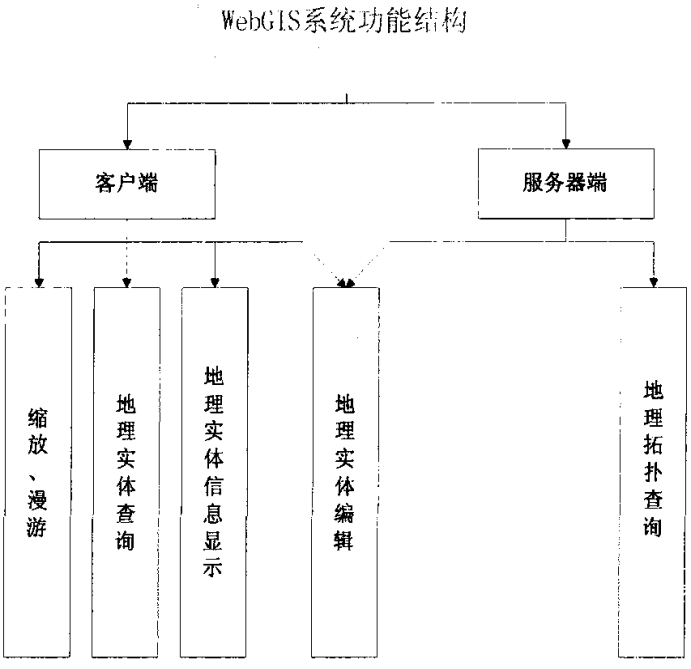


图 11. WebGIS 系统功能结构

### 4.2 WebGIS 系统结构

WebGIS 原型系统的设计采用 B/S 结构和 MVC 模型。B/S 结构即是以浏览器和服务器分别为客户端和服务器端的网络服务结构。MVC 模型是将网络服务从功能结构上分成三层，即模型层(Model)、表现层(View)和控制层(Control)。MVC 是一种常用的 Web 设计模式。这种模式很好得分离业务逻辑层和表示逻辑层，有效的整合各种 Web 程序[4][5]。

控制层(Controller)定义应用程序的行为，客户端请求被控制层分发或操作重定向。 ActionServlet 接受到客户端的请求以后，将调用模型层的组件进行数据操作。

模型层(Model)负责表达和访问数据，执行商业逻辑操作。模型层接受控制层的调用，并将操作结果返回给控制层。

视图层(View)负责内容的显示，接受控制层的重定向，或者将控制层收到的结果进行显示，同时还负责部分将客户端请求传送给控制层的工作。

MVC 模式示意图如下(图 11):

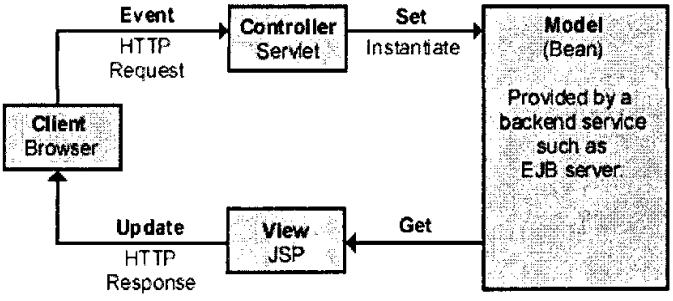


**图 12. MVC 模式示意图**

采用 B/S 架构和 MVC 模式，WebGIS 系统将包括三部分组成。

客户端：一般为 IE 浏览器。客户端负责地图显示，地图操作，包括地图显示接口，用户操作结构。一般用 HTML 文件，Javascript 文件和 JSP 文件组成。

HTML 文件负责页面控制，JSP 负责数据传输，Javascript 负责用户操作接口实现。部分 WebGIS 功能组件将由 Javascript 文件组成。

服务器端：采用 Tomcat 服务器。使用 Servlet 程序、JavaBean 组件实现服务器的功能。Tomcat 是一款支持 JSP 的开源网络服务器，特点是灵活，部署方便，易于管理，支持跨平台，移植性非常好。WebGIS 部分复杂功能组件以 ESRI 公司的 MapObject-JavaEdtion 为基础进行二次开发。

数据服务：地理数据和属性数据存放在 Oracle 数据库中，并采用 ESRI 公司的 ArcSDE 软件作为空间数据库引擎。通过 ArcSDE，WebGIS 组建可以方便的访问空间数据库。ArcCatalog 等平台软件也可以通过 ArcSDE 进行空间数据的存取。

本 WebGIS 系统为分布式应用，网络服务器和数据服务分别存放在不同的计算机上，平衡了服务器的负担，同时也方便进行管理。

本 WebGIS 系统的结构图如下：

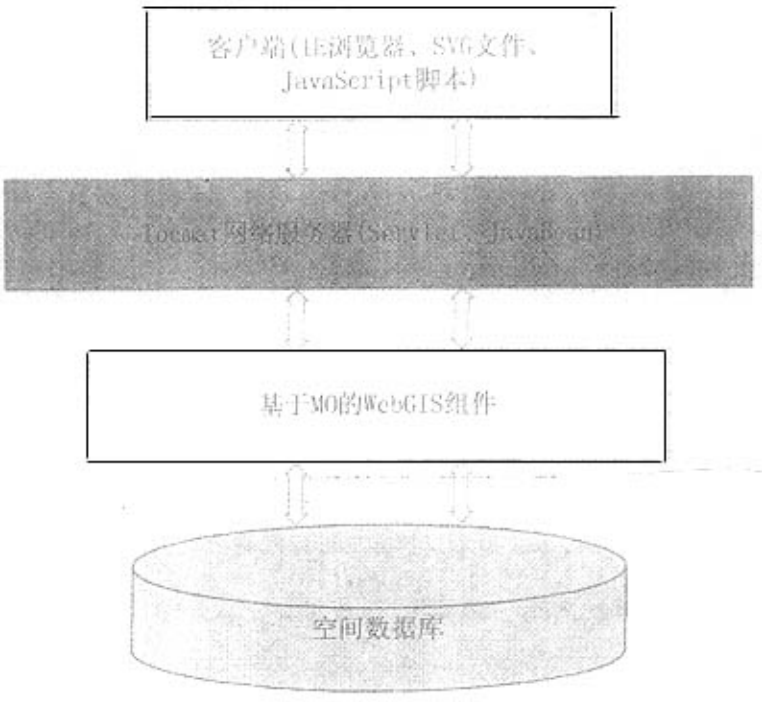


图 13. WebGIS 系统结构

### 4.3 WebGIS 系统建立

**1．软件采用：**

编程语言采用 Java。Java 在网络设计中具有很多优点，如简单、面向对象、分布式、解释性、可靠、安全、结构中立性、可移植性、高性能、多线程、动态性等。Java 可以运行与任何微处理器，用 Java 开发的程序可以在网络上传输，并运行于任何客户机上。

服务器采用 Tomcat 服务器，这种服务器对 JSP 支持良好，同时可以同 IIS，Apache 等本身不能支持 JSP 的网络服务器协同作业。Tomcat 服务器比较小巧，部署和使用都非常方便。

WebGIS 功能组件采用 MapObject 作为二次开发组件。MapObject 功能强大，能够进行各种空间数据操作。

空间数据库采用 Oracle 数据库，空间数据引擎使用 ArcSDE。Oracle 数据库对空间数据的支持是目前商业数据库中比较好的，尤其擅长对海量数据进行存储和处理。

**2．数据来源：**

数据来自 shp 格式的深圳地图文件。

**3．数据处理：**

使用 ArcCatalog 工具将 shp 文件通过 ArcSDE 导入 Oracle 空间数据库中。基于 MapObject，使用 Java 编程语言开发了空间数据转换 SVG 文件的组件。空间数据库更新的同时调用该组件更新 SVG 地图的内容，保持客户端与服务器端数据一致性。

## 4.4 WebGIS 系统功能实现及展示

**1、缩放、漫游**

缩放和漫游是 WebGIS 的基本功能。实现这个功能的核心在于改变地图的显示范围。在 SVG 中，svg 标记的 viewBox 属性是控制地图显示范围的。因此只要根据用户的操作要求，改变 viewBox 的值就可以了。漫游不同于缩放的地方在于它只是改变地图显示的中心，而不用改变显示范围的大小。只要将 viewBox 属性的代表左上角的点坐标改变就可以了。用 Javascript 在客户端去计算地图的最大范围是耗时而繁琐的，显示全图功能要求客户端记住地图的最大显示范围。

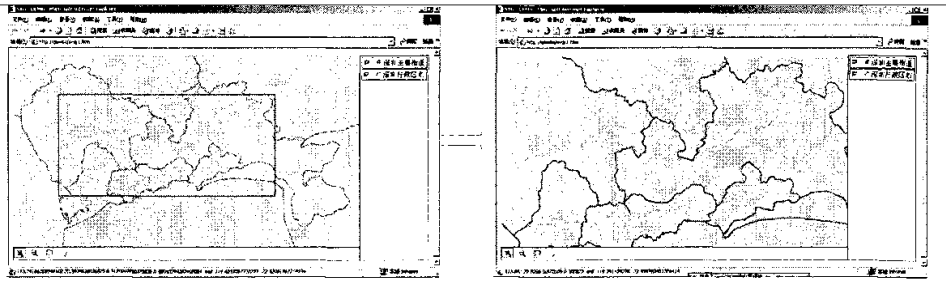

图 14. 地图放大

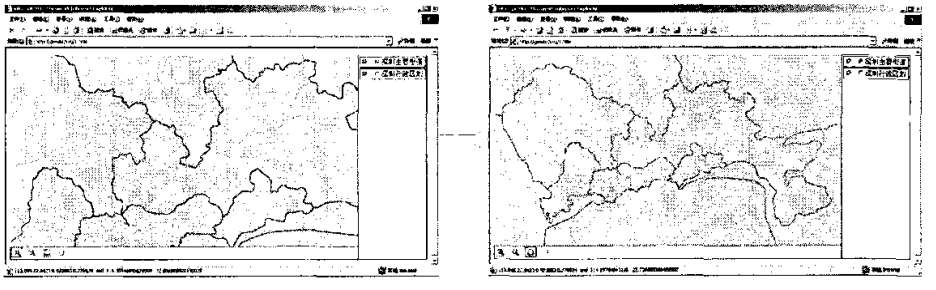

图 15. 显示全图

**2、控制图层显示**

控制图层显示的功能非常容易实现。任何一个图形对象都具有自己的风格

(style)。其 style 具有 visibility 属性，如果将这个属性设置为 hidden，这个图形就会被隐藏。本系统的 SVG 地图中，每个图层是用 g 来标记的，只要改变这个标记图层的 g 的 visibility 属性，就可以控制图层的显示与隐藏。在图 16 中演示了这个功能。

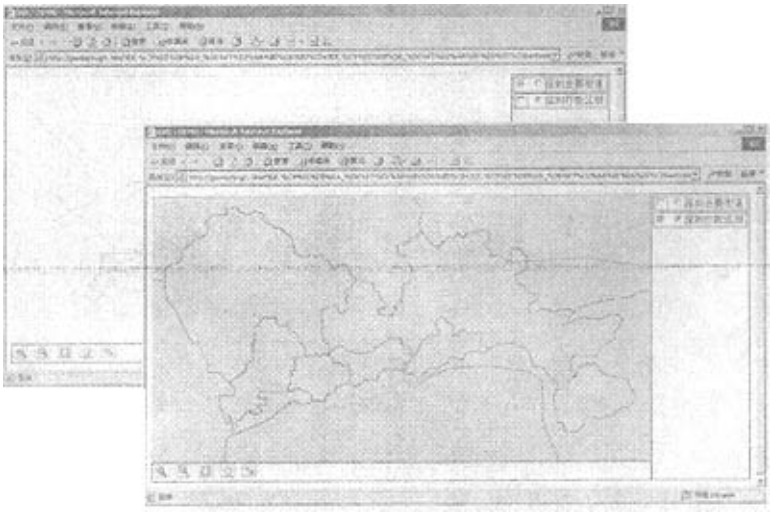

图 16. 控制图层显示

### 3、显示地理实体基本属性

SVG 图形可以触发事件。使用 addEventListener 方法，可以给任何图形对象加上诸如鼠标点击、鼠标移动等动作监听。下面是加入鼠标事件的代码：

```
obj.addEventListener("mouseover",doMouseOver(),false);
function doMouseOver(evt) {
    var obj = evt.target;
    ...
}
```

在上面的代码中，图形对象被加上了鼠标移上的动作，通过这个动作可以得到鼠标下方的对象，从而可以得到这个对象的全部信息。图 17 是显示属性的效果图。

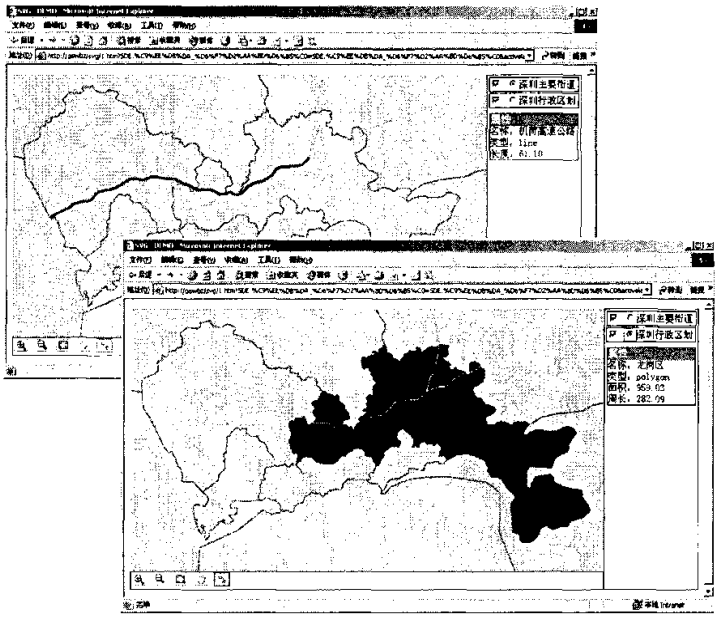

图 17. 显示属性

### 4、地理实体查询

地理实体的查询包括好多方式，通过名字、周长、面积等等。本系统将地理实体的基本属性都作为节点属性放入 SVG 地图文件中。只要遍历 SVG 中所有的节点，取出相应属性进行判断，就可以将符合查询条件的地理实体过滤出来。

本试验系统将地理实体的名字作为 ID 放入了 SVG 地图文件，因此对地理实体名字的查询很简单，只要使用 getElementById 方法就可以将符合查询条件的地理实体找到。

图 18 是地理实体查询的效果图。

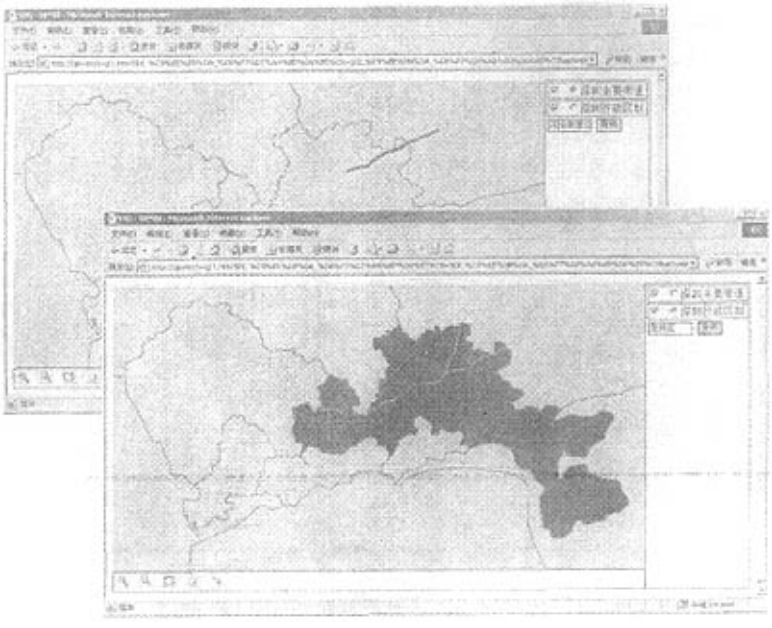

图 18. 地理实体查询

### 5、地理实体编辑

地理实体的编辑是比较复杂的功能。

删除和复制功能只需要对操作对象进行 removeChild 和 appendChild 操作就可以了。但是在客户端改变操作对象(图形元素)需要对该对象的每个点进行解析、变换，达到编辑的效果。本试验系统仅对地理实体的平移做了试验。

编辑的结果通过服务器端的 WebGIS 功能组件写在服务器的空间数据库中，并实时更新 SVG 地图文件。

图 19. 地理实体删除

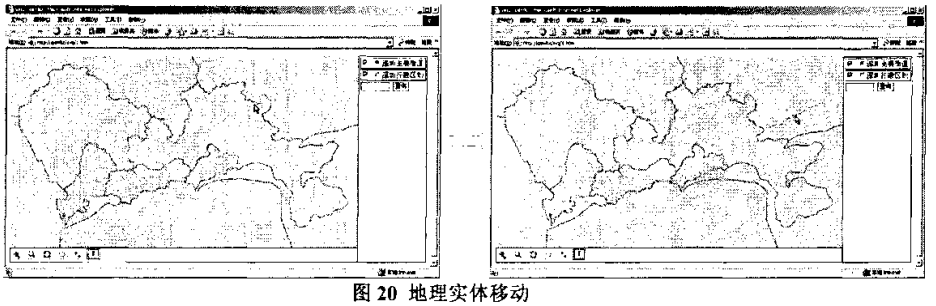

图 20 地理实体移动

### 6、地理实体拓扑查询

目前直接利用 SVG 的空间数据对 SVG 地图进行拓扑分析是比较复杂的事情，在客户端实现比较困难。大多数的 SVG 图形的拓扑分析都是在服务器上进行。ESRI 功能的 Map Object 组件非常适合做这个事情。只需将分析条件(框选、线选)以及目标图层等参数传到服务器端，服务器上的 WebGIS 功能组件就可以将符合拓扑条件的地理实体检索出来，并将其具体信息传到客户端，供 SVG 地图显示。

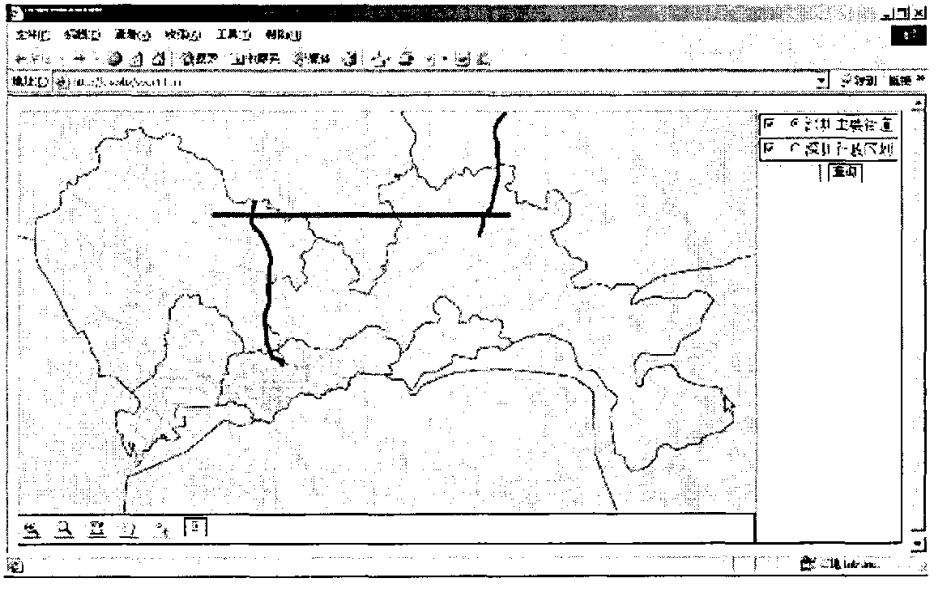

图 21 地理实体拓扑查询

## 4.4 基于 SVG 的 WebGIS 与基于 ArcIMS 的 IMAGE 服务系统的比较

基于 ArcIMS 的 IMAGE 服务系统是目前比较流行的瘦客户端的 WebGIS 系统。这种系统的 WebGIS 功能组件全部安装在 WebGIS 服务器上。客户端提交请求，这些请求被定制为 ArcXML 代码发送到 ArcIMS 服务器上，由服务器进行解

析。服务器对用户请求的相应仍然以 ArcXML 的形式返回，由服务器段的 Web 组件或者客户端的 Javascript 脚本进行解析，将结果以图片和其他形式显示在浏览器上。

这种服务的优点在于：

1、服务与浏览器平台无关，只要可以连接上网络，就能够使用该 WebGIS 服务。

2、用户不用下载任何插件、控件就可以进行地理数据的下载和使用。

3、服务器可以安装在分布式的计算机网络内，分散网络访问的负担。

4、该服务提供了绝大多数的查询功能，能够满足一般地理数据浏览和查询的需要。

这种服务也有自己的缺陷：

1、客户端数据的刷新要靠服务器生成新的图片传递到浏览器来实现。这加重了服务器的负担，也占用了大量的网络带宽。

2、服务不能提供最基本的地理数据编辑功能，对于实现 B/S 结构的完整意义上的 WebGIS 是一个很大的缺陷。

3、用户的对地理数据的访问都需要从客户端搜集用户操作信息，然后返回服务器进行刷新，浪费了大量的时间。

与此比较，基于 SVG 的 WebGIS 系统具有非常大的优势。SVG 可以采用与 ArcIMS 的 IMAGE 服务相同的体系架构，但是在以下几个方面将远远胜过 ArcIMS 的 IMAGE 服务。

1、SVG 只需要下载一次，此后只需要服务器与客户端保持同步更新即可。节省了大量的资源。

2、SVG 可以在客户端做各种各样的图形操作，完全可以达到与桌面 GIS 相同的功能效果。

3、SVG 可以存储大量的地理信息、属性信息，因此对一些基本信息的搜索只需在客户端即可完成。

4、SVG 是矢量图形，对其缩放同样只需在客户端完成。

5、SVG 可以实现多种多样的专题图效果，而实现同等效果的图片专题图就要耗费很大的资源和时间。

只是目前大部分浏览器尚不支持 SVG 图形，需要下载插件，但 SVG 给

WebGIS 带来的更大便利，而且随着 SVG 网络图形的广泛使用，目前许多厂商开始发展 SVG 插件、SVG 浏览器，也有众多的基于 SVG 的 WebGIS 系统在网络上建立起来。相信不久的将来，SVG 完全可以做到象 Flash 一样，占领网络图形市场。

# 第五章 结论与展望

基于 SVG 的 WebGIS 系统，充分发挥了 SVG 图形文件的优点。

SVG 良好的文档可操控性，对 WebGIS 在客户端对地图的缩放、漫游、搜索等操作提供了极大的方便，节省了使用者大量的时间，同时开发人员的工作也变得非常简单。

SVG 丰富图形和色彩，使 WebGIS 客户端在地图现实方面取得良好的效果，任何缩放程度下都可以获得清晰的分辨力和打印效果。

SVG 具有相当小的文件尺寸，比起瘦客户端 JPEG 图片的传输、胖客户端地理数据和控件的打包传输都具有无可比拟的速度优势。同时 SVG 文件一旦下载下来，很少需要重新下载，节约了网络带宽，减轻了网络负担。

SVG 完全基于文本，易于编辑，相比较 Flash 等其他矢量图形，又具有易于修改的特性。

SVG 文件是基于 XML 标准的，对于统一地理数据格式，共享地理数据有非凡的意义。

目前 SVG 的应用还不是特别繁荣，SVG 在浏览器端的显示还需要安装插件，但是有生命力的新生事物必将迅速发展，茁壮成长，并在网络图形以及 WebGIS 领域起到积极、促进的作用，产生广泛、深远的影响。

# 参考文献

1.邬伦,刘瑜等.地理信息系统——原理、方法和应用[M].北京:科学出版社,2001.314-316

2.王继周，付俊娥，李成名，朱欣焰. 基于 J2EE 的 WebGIS 应用服务器构建技术研究[J]. 计算机工程与应用，2003;(28):36-38.

3.肖国强,冯燕.一个基于 Java/J2EE 的 WebGIS 的模型研究.计算机应用研究，2003;(5):110-112.

4.何成万，余秋惠.MVC 模型 2 及软件框架 Struts 的研究.计算机工程，2002;(6):274-275.

5.寇毅,吴力文.基于 MVC 设计模式的 Struts 框架的应用方法.计算机应用，2003;(11):91-93.

6.胡金星,潘懋. 基于 0racle Spatial 的 WebGIS 解决方案.计算机工程与应用，2003;(5):184-186.

7. W3C, 2003, Scalable Vector Graphics (SVG) 1.1 Specification
http://www.w3.org/TR/SVG/

8. About SAX http://www.saxproject.org/

9. Mobile GIS http://www.trimble.com/mgis_mobilegis.shtml

10. Scalable Vector Graphics (SVG) http://www.w3.org/Graphics/SVG/

11.Adobe, 2003, SVG Zone http://www.adobe.com/svg

12.王兴玲. SVG 与矢量地图的 Web 发布技术[J]. 计算机工程与应用, 2002 , 10 (10) .

13.孙少红, 边馥苓. SVG 网络图像标准支持下的地图显示[J ] . 测绘信息与工程, 2002 , 27 (5) .

14.周文生. 基于 SVG 的 WebGIS 研究[J ] . 中国图形图像学报, 2002 , 7(7) .

# 附录 A　WebGIS 试验系统代码

```
<title>SVG - DEMO</title>
<style>
    <!--
        input{border:'solid 1px white'}
    -->
</style>
<body onload="fun()">
<table border="0" align="center" style="table-layout:fixed" cellpading="0" cellspacing="0">
<tr>
<td width="800">
<embed name="map" align="absbottom" pluginspage=http://www.adobe.com/svg/viewer/install/
src="test.svg" width="800px" height="310px" type="image/svg+xml">
</td>
<td valign="top" rowspan="2" style="border:'solid black 1px'">
<form name="mapcontrol">
<table style="table-layout:fixed">
<tr>
<td style="border:'solid black 1px'">
<input type="checkbox" name="SDE.深圳_主要街道" value="SDE.深圳_主要街道" checked
onClick="visControl(this)">
<input type="radio" checked name="activelayer" value="SDE.深圳_主要街道"
onclick="makeact(this)">深圳主要街道
</td>
</tr>
<tr>
<td style="border:'solid black 1px'">
<input type="checkbox" name="SDE.深圳_背景" value="SDE.深圳_背景" checked
onClick="visControl(this)">
<input type="radio" name="activelayer" value="SDE.深圳_背景" onclick="makeact(this)">深圳
行政区划
</td>
</tr>
<tr>
<td>
<form name="query">
<input type="text" name="field" style="border:'solid 1px black'" size="8">
<input type="button" onclick="query(this.form.field.value);" value="查询" style="border:'solid
1px black'">
</form>
</td>
</tr>
<tr>
<td id="x">
</td>
</tr>
</table>
</from>
</td>
</tr>
<tr>
<td style="border:'solid black 1px';">
<input type="image" src="images/zoomin.gif" onclick="doFunction('zoomin',this);return false"/>
```

```
<input type="image" src="images/zoomout.gif" onclick="doFunction('zoomout',this);return
false;"/>
<input type="image" src="images/zoomfull.gif" onclick="doFunction('zoomfull',this);return
false;"/>
<input type="image" src="images/pan.gif" onclick="doFunction('pan',this);return false;"/>
<input type="image" src="images/query.gif" onclick="doFunction('showtip',this);return false;"/>
<input type="image" src="images/edit.gif" onclick="doFunction('edit',this);return false;"/>
</td>
</table>
</body>
<script language="javascript">
    var svg;
    var actlayer;
    var objinfo;
    var buttonClicked;
    var command;
    var rect;
    var bmousedown = false;
    var mask;
    var times;
    var initviewBox;
    var initTransform;
    var editTarget = null;
    var x1;
    var y1;
    function fun() {
        svg = map.getSVGDocument();

        this.UpdateMenu = function(menus)
        {
         var ver = this.map.window.getSVGViewerVersion();
         if(ver.indexOf("Adobe")!=-1)
         {
          var newMenu=this.map.window.parseXML(menus,this.map.window.contextMenu);

          if(newMenu.nodeName.toUpperCase() == "MENU")
          //if(ver.indexOf("6.0")==-1)
          {

this.map.window.contextMenu.replaceChild(newMenu,this.map.window.contextMenu.firstChild);
           alert();
          }
          else
          {

this.map.window.contextMenu.replaceChild(newMenu,this.map.window.contextMenu.firstChild);
          }
         }
        }


          ms = "<menu>";
          ms = ms + "<header>菜单</header>";
          ms = ms + "<item onactivate=\"doFunction('move',null)\">移动(M)</item>";

          ms = ms + "<item onactivate=\"doFunction('copy',null)\">复制(C)</item>";
          ms = ms + "<item onactivate=\"doFunction('cut',null)\">剪切(X)</item>";
```

```
            ms = ms + "<item onactivate=\"doFunction('delete',null)\">删除(D)</item>";
            ms = ms + "</menu>";

        UpdateMenu(ms)



        var viewbox = svg.getElementById("mainview").getAttribute("viewBox");
        initviewBox = viewbox;
        initTransform = svg.rootElement.getChildNodes.item(0).getAttribute("transform");
        var s = viewbox.split(" ");

        times = s[2]/s[3];
        map.height = map.width/times;
        c = svg.rootElement.getChildNodes.item(0).getChildNodes:
        for(var i = 0; i < c.length; i++) {
            z = c.item(i).getChildNodes();
            for(var j = 0; j < z.length; j++) {
                o = z.item(j);
                o.addEventListener("mouseover",over,false);
                o.addEventListener("mousedown",down,false);
                o.addEventListener("mousemove",move,false);
                o.addEventListener("mouseup",up,false):
                o.addEventListener("mouseclick",click,false):
                o.addEventListener("mouseout",out,false):
            }
        }
        for(var i = 0; i < document.mapcontrol.activelayer.length; i++) {
            if(document.mapcontrol.activelayer[i].checked) {
                actlayer = document.mapcontrol.activelayer[i].value;
            }
        }
    }
    function over(evt) {
        var obj = evt.target;
        if(obj.parentNode.id != actlayer){
            return;
        }
        else if(command != "showtip") {
            return;
        }
        else {
            var text = "<div style='border:solid gray
1px;background-color:gray;color:white;align:center'><b>属性:</b></div>";
            text += "名称：" + obj.id + "<br>";
            text += "类型：" + obj.parentNode.getAttribute("type") + "<br>";
            if(obj.parentNode.getAttribute("type") =="polygon") {
            obj.setAttribute("prefill",obj.getAttribute("fill")):
            obj.setAttribute("fill","blue");
            text += "面积：" + parseFloat(obj.getAttribute("area")).toFixed(2) + "<br>";
            text += "周长：" + parseFloat(obj.getAttribute("length")).toFixed(2) + "<br>";
            }
            else if(obj.parentNode.getAttribute("type") == "line") {
            text += "长度：" + parseFloat(obj.getAttribute("length")).toFixed(2) + "<br>";
            obj.setAttribute("prestroke-width",obj.getAttribute("stroke-width")):
            obj.setAttribute("stroke-width",0.005);
```

```
            }
            obj.setAttribute("prestroke",obj.getAttribute("stroke"));
            obj.setAttribute("stroke","red");
            var j = document.createElement("<div>");
            j.innerHTML=text;
            j.style.color = "blue";
            j.style.borderWidth=1;
            j.style.border = "solid black 1px";
            j.style.position="relative";
            j.style.pixelTop=5;
            if(document.getElementById('x').lastChild != null) {

document.getElementById("x").replaceChild(j,document.getElementById('x').lastChild);
            }
            else {
                document.getElementById("x").appendChild(j);
            }
        }
}
function down(evt) {
        var obj = evt.target;
        var xdown = evt.getClientX();
        var ydown = evt.getClientY();
        xdown = transPixelToUser("x",xdown);
        ydown = transPixelToUser("y",ydown);
        x1 = xdown;
        y1 = ydown;
        bmousedown = true;
        if(command == "zoomin") {
            rect = svg.createElement("rect");
            rect.setAttribute("x1",xdown);
            rect.setAttribute("y1",ydown);
            rect.setAttribute("fill","none");
            rect.setAttribute("stroke","blue");
            rect.setAttribute("stroke-width","0.002")
            rect.getStyle.setProperty("z-index",1);
            svg.rootElement.getChildNodes.item(0).insertBefore(rect,mask);
        }
        else if(command == "edit") {
            if(editTarget != null) {
                editTarget.setAttribute("stroke",editTarget.getAttribute("prestroke"));


editTarget.setAttribute("stroke-width",editTarget.getAttribute("prestroke-width"));
                if(editTarget.parentNode.getAttribute("type") == "polygon") {
                    editTarget.setAttribute("fill",editTarget.getAttribute("prefill"));
                }
            }
            if(obj.parentNode.getAttribute("id") == actlayer) {
                obj.setAttribute("prestroke",obj.getAttribute("stroke"));
                obj.setAttribute("prestroke-width",obj.getAttribute("stroke-width"));
                obj.setAttribute("stroke","blue");
                obj.setAttribute("stroke-width",0.005);
                if(obj.parentNode.getAttribute("type") == "polygon") {
                    obj.setAttribute("prefill",obj.getAttribute("fill"));
                    obj.setAttribute("fill","blue");
                }
```

```
                editTarget = obj;
            }
            else {
                return;
            }
        }
    }
    function move(evt) {
        var obj = evt.target;
        var xmove = evt.getClientX();
        var ymove = evt.getClientY();
        xmove = transPixelToUser("x",xmove);
        ymove = transPixelToUser("y",ymove);
        if(bmousedown ==false) {
            return;
        }
        var obj = evt.target;
        var xmove = evt.getClientX();
        var ymove = evt.getClientY();
        xmove = transPixelToUser("x",xmove);
        ymove = transPixelToUser("y",ymove);
        if(command == "zoomin") {
            var xdown = rect.getAttribute("x1");
            var ydown = rect.getAttribute("y1");
            var x = xmove < xdown ? xmove : xdown;
            var y = ymove < ydown ? ymove : ydown;
            var width = Math.abs(xdown - xmove);
            var height = Math.abs(ydown - ymove);
            rect.setAttribute("x",x);
            rect.setAttribute("y",y);
            rect.setAttribute("width",width);
            rect.setAttribute("height",height);

        }
        else if(command == "edit") {
            if(editTarget == null) {
                return;
            }
            else if(editTarget != obj) {
                return;
            }
            else {
                var disx = xmove - x1;
                var disy = ymove - y1;
                var gTransform = svg.createElement("g");
                gTransform.setAttribute("transform","translate(" + disx + "," + disy + ")");
                gTransform.setAttribute("stroke-width",0.001);
                svg.rootElement.getChildNodes.item(0).appendChild(gTransform);
                status = gTransform.getAttribute("transform");
                gTransform.appendChild(editTarget);
            }
        }
    }
    function up(evt) {
        if(bmousedown == false) {
            return;
        }
```

```
            bmousedown = false;
            var obj = evt.target;
            if(command == "zoomin") {
                var s = svg.rootElement.getAttribute("viewBox").split(" ");
                if(rect.getAttribute("width") / rect.getAttribute("height") > times) {
                    var height = rect.getAttribute("width") / times;
                    rect.setAttribute("height",height);
                }
                else {
                    var width = rect.getAttribute("height") * times;
                    rect.setAttribute("width",width);
                }
                var viewbox = rect.getAttribute("x") + " " + rect.getAttribute("y") + " " +
rect.getAttribute("width") + " " + rect.getAttribute("height");
                svg.rootElement.setAttribute("viewBox",viewbox);
                s = viewbox.split(" ");
                var translate = 2 * s[1] + parseFloat(s[3]);
                svg.rootElement.getChildNodes.item(0).setAttribute("transform","matrix(1 0 0 -1 0
" + translate + ")");
                svg.rootElement.getChildNodes.item(0).removeChild(rect);

            }
        }
        function click(evt) {
            var obj = evt.target;
            alert();
            if(command == "edit") {
                alert();
                if(obj.parentNode.getAttribute("id") == actlayer) {
                    obj.setAttribute("stroke","blue");
                    obj.setAttribute("stroke-widt",0.005);
                    if(obj.parentNode.getAttribute("type") == "polygon") {
                        obj.setAttribute("fill","blue");
                    }
                    editTarget = obj;
                }
                else {
                    return;
                }
            }
        }
        function out(evt) {
            var obj = evt.target;

            if(command == "zoomin") {
                if(bmousedown == false) {
                    return;
                }
                bmousedown = false;
                var s = svg.rootElement.getAttribute("viewBox").split(" ");
                if(rect.getAttribute("width") / rect.getAttribute("height") > times) {
                    var height = rect.getAttribute("width") / times;
                    rect.setAttribute("height",height);
                }
                else {
                    var width = rect.getAttribute("height") * times;
                    rect.setAttribute("width",width);
```

```
            }

            var viewbox = rect.getAttribute("x") + " " + rect.getAttribute("y") + " " +
rect.getAttribute("width") + " " + rect.getAttribute("height");
            svg.rootElement.setAttribute("viewBox",viewbox);
            s = viewbox.split(" ");
            var translate = 2 * s[1] + parseFloat(s[3]);
            svg.rootElement.getChildNodes.item(0).setAttribute("transform","matrix(1 0 0 -1 0
" + translate + ")");
            svg.rootElement.getChildNodes.item(0).removeChild(rect);
        }
        else if(command == "showtip") {
            if(obj.parentNode.id != actlayer){
                return;
            }
            if(obj.parentNode.getAttribute("type") == "polygon") {
                obj.setAttribute("fill",obj.getAttribute("prefill"));
            }
            else if(obj.parentNode.getAttribute("type") == "line") {
            obj.setAttribute("stroke-width",obj.getAttribute("prestroke-width"));
            }
            obj.setAttribute("stroke",obj.getAttribute("prestroke"));
        }


    }
    function visControl(obj) {
        if(obj.checked) {
            var style = svg.getElementById(obj.value).getStyle();
            style.setProperty("visibility","visible");
        }
        else {
            var style = svg.getElementById(obj.value).getStyle();
            style.setProperty("visibility","hidden");
        }
    }
    function makeact(obj) {
        actlayer = obj.value;
    }
    function doFunction(fun,obj) {
        if(command != "edit") {
            command = fun;
        }
        if(command == "zoomin" || command == "zoomout") {
            mask = svg.createElement("rect");
            var s = svg.rootElement.getAttribute("viewBox").split(" ");
            mask.setAttribute("x",s[0]);
            mask.setAttribute("y",s[1]);
            mask.setAttribute("width",s[2]);
            mask.setAttribute("height",s[3]);
            mask.setAttribute("stroke","red");
            mask.setAttribute("stroke-width",0.002);
            mask.setAttribute("fill","blue");
            var style = mask.getStyle();
            style.setProperty("opacity",0);
            svg.rootElement.getChildNodes.item(0).appendChild(mask);
            mask.addEventListener("mousedown",down,false);
```

```
            mask.addEventListener("mousemove",move,false);
            mask.addEventListener("mouseup",up,false);
            mask.addEventListener("mouseout",out,false);
        }
        else if(command == "zoomfull") {
            svg.rootElement.setAttribute("viewBox",initviewBox);
            svg.rootElement.getChildNodes.item(0).setAttribute("transform",initTransform);
            obj.blur();
            return;
        }
        else if(command == "showtip") {
            var childs = svg.rootElement.getChildNodes.item(0).getChildNodes;
            for(i = 0; i < childs.length; i++) {
                if(childs.item(i) == mask) {
                    svg.rootElement.getChildNodes.item(0).removeChild(mask);
                }
            }
        }
        else if(command == "edit") {
            var childs = svg.rootElement.getChildNodes.item(0).getChildNodes;
            for(i = 0; i < childs.length; i++) {
                if(childs.item(i) == mask) {
                    svg.rootElement.getChildNodes.item(0).removeChild(mask);
                }
            }
            if(fun == "delete") {
                if(editTarget == null) {
                    return;
                }
                editTarget.parentNode.removeChild(editTarget);
                editTarget = null;
            }

        }
        if(buttonClicked != null) {
            buttonClicked.style.border="white 1px solid";
        }
        if(obj == null) {
            return;
        }
        buttonClicked = obj;
        obj.style.border="red 1px solid";
        obj.blur();

}
function zoomin() {
    var x = transPixelToUser("y",0);
    alert(x);
}
function transPixelToUser(x,data) {
    var s = svg.rootElement.getAttribute("viewBox").split(" ");
    var scalex = map.width / s[2];
    var scaley = map.height / s[3];
    if(x=="x") {
        data = parseFloat(s[0]) + data / scalex;
        return data;
    }
```

```
            else if(x == "y") {
                data = parseFloat(s[1]) - data / scaley + parseFloat(s[3]) ;
                return data;
            }
        }
        function query(name) {
            var obj = svg.getElementById(name);
            if(obj == null) {
                alert("没有找到对象！ ");
                return;
            }
            if(obj.parentNode.getAttribute("id") != actlayer) {
                alert("在当前层未找到对象！ ");
                return;
            }
            obj.setAttribute("stroke","red");
            if(obj.parentNode.getAttribute("type") == "polygon") {
                obj.setAttribute("fill","blue");
            }
            obj.setAttribute("stroke-width","0.005");
        }
</script>
```

服务器端 SVG 图形转换组件

```
package com.seagolden.app;

import com.esri.mo2.client.sde.*;
import com.esri.mo2.src.sys.*;
import com.esri.mo2.map.dpy.*;
import com.esri.mo2.data.feat.*;
import com.esri.mo2.cs.geom.*;
import java.io.File;
import java.io.FileWriter;
import java.io.BufferedWriter;
import com.esri.mo2.util.Units;

public class SdeTest {
  public SdeTest() {
    try {
      File f = new File("c:\\test.svg");
      FileWriter fw = new FileWriter(f);
      BufferedWriter bw = new BufferedWriter(fw);
      String s = "<?xml version=\"1.0\" standalone=\"no\" ?>";
      bw.write(s);

      s = "<!DOCTYPE svg PUBLIC \"-//W3C//DTD SVG 1.0//EN\"
\"http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd\">";
      bw.write(s);


      SdeConnection sdeConnection = new SdeConnection("210.77.93.245", 5151,
          "",
          "sde", "123");
      Source source = SourceCache.getSource(sdeConnection);

      source.connect();
      Content[] contents = source.getRoots();
```

```
int rownum = 0;
for (int i = 0; i < contents.length; i++) {

   Content c = contents[i];
   if (c instanceof BaseFolder) {
      BaseFolder folder = (BaseFolder) c;
      com.esri.mo2.data.feat.FeatureDataset fd = (com.esri.mo2.data.feat.
            FeatureDataset) folder.getData(
            "com.esri.mo2.data.feat.FeatureDataset");
      FeatureClass[] fcs = fd.getFeatureClasses();
      Cursor cursor1 = fcs[0].getSelectedData(new BaseSelectionSet(fcs[0]));
      Data d = cursor1.next();

      Content[] c2 = folder.getContents();

      if (c2.length > 0) {
         Content layerContent = c2[0];
         FeatureLayer fLayer = (FeatureLayer) layerContent.getData(
               FeatureLayer.class.getName());

         double minx = fLayer.getExtent().getMinX();
         double miny = fLayer.getExtent().getMinY();
         double width = fLayer.getExtent().getWidth();
         double height = fLayer.getExtent().getHeight();
         s = "<svg viewBox=\"" + minx + " " + miny + " " + width + " " +
               height + "\" xmlns=\"http://www.w3.org/2000/svg\">";
         bw.write(s);


         FeatureClass fc1 = fLayer.getFeatureClass();
         if (fc1.getFeatureType() == FeatureClass.POLYGON) {
            s = "<g id=\"" + fLayer.getName() + "\"";
            s += " type=\"polygon\">";
         }
         else if (fc1.getFeatureType() == FeatureClass.LINE) {
            s = "<g id=\"" + fLayer.getName() + "\"";
            s += " type=\"line\">";
         }
         else {
            continue;
         }
         bw.write(s);


         Fields fields = fc1.getFields();
         BaseQueryFilter bqf = new BaseQueryFilter();
         bqf.setSubFields(fields);
         String sqlStr = fields.getNames()[1] + " <> 'xyz'";
         bqf.setWhereClause(sqlStr);
         bqf.setUseCache(false);
         Cursor cur = fc1.search(bqf);
         while (cur.hasMore()) {
            Feature feature = (Feature) cur.next();
            FeatureGeometry fg = feature.getGeometry();
            if (fg instanceof Polyline) {
               Polyline pl = (Polyline) fg;
               String name = feature.getValue(1).toString();
```

```
                String objectId = feature.getValue(0).toString();
                double length = pl.getPerimeter();
                length = Units.convert(length, Units.DECIMAL_DEGREES,
                                        Units.KILOMETERS);
                s = "<path";
                s += " id=\"" + name + "\"";
                s += " objectid=\"" + objectId + "\"";
                s += " length=\"" + length + "\"";
                s += " fill=\"none\" stroke=\"orange\" d=\"";
                Path[] paths = pl.getPaths();
                for (int j = 0; j < paths.length; j++) {
                  PointCollection pc = paths[j].getPoints();
                  PointIterator pi = pc.iterator();
                  int m = 0;
                  while (pi.hasNext()) {
                    Point p = new Point();
                    pi.nextPoint(p);
                    if (m == 0) {
                      s += "M";
                    }
                    else {
                      s += "L";
                    }
                    s += p.getX() + "," + p.getY() + " ";
                    m++;
                  }
                }
                s += "\"/>";
                bw.write(s);


              }

              if (fg instanceof Polygon) {
                Polygon polygon = (Polygon) fg;
                Ring[] rings = polygon.getRings();
                String name = feature.getValue(1).toString();
                String objectId = feature.getValue(0).toString();
                double length = polygon.getOutsidePerimeter();
                double area = polygon.getArea();
                length = Units.convert(length, Units.DECIMAL_DEGREES,
                                        Units.KILOMETERS);
                double times = Math.pow(length,2) /
Math.pow(polygon.getOutsidePerimeter(),2);
                area = area * times;
                s = "<path";
                s += " id=\"" + name + "\"";
                s += " objectid=\"" + objectId + "\"";
                s += " length=\"" + length + "\"";
                s += " area=\"" + area + "\"";
                s += " fill=\"gray\" stroke=\"black\" d=\"";
                for (int n = 0; n < rings.length; n++) {
                  PointCollection pc = rings[n].getPoints();
                  PointIterator pi = pc.iterator();
                  int m = 0;
                  while (pi.hasNext()) {
                    Point p = new Point();
```

```
                        pi.nextPoint(p);
                        if (m == 0) {
                            s += "M";
                        }
                        else {
                            s += "L";
                        }
                        s += p.getX() + "," + p.getY() + " ";
                        m++;
                    }
                }
                s += "\"/>";
                bw.write(s);


            }
        }
        }
        s = "</g>";
        bw.write(s);
    }
    }
    s = "</svg>";
    bw.write(s);
    source.disconnect();
    bw.flush();
    bw.close();
    fw.close();
}
catch (Exception ex) {
    ex.printStackTrace();
}
}
}
```

## 已发表和待发表的论文

1、《WebGIS 客户端实现技术的比较研究及应用》(待刊) 吴彬卓,夏斌 《农机化研究》2006 年第 3 期发表（2006 年 3 月 20 日出版）。

2、Mobile Phone GIS Based on Mobile SVG(待刊) 吴彬卓,夏斌 IGARSS2005 2005 年 7 月发表。会议网址：http://www.igarss05.org。

# 致谢

时光荏苒，三年的研究生生活就要结束了。从北京研究生院回到广州地化所，就投入了 GIS 实验室枯燥、繁忙的学习中。每天电脑屏幕和花花绿绿的代码是很辛苦的，但为了能够提高自身的水平，拓宽自身的知识，掌握最新最实用的技术，每天学习到深夜是家常便饭。虽然比较辛苦，但是在导师的关怀下，师兄弟们的帮助和鼓励下，依然觉得生活过得非常的充实。站在学生生涯的终点和社会的起点上，我对过去挥挥衣袖，没有一丝遗憾，我对未来招招手，充满了憧憬与期待。

身上装载着地化所赋予的知识和技能，我将投入到为祖国、社会做直接贡献的工作中去，用辛勤的工作，诚实的劳动回报地化所的培养，导师的指导和各位同门的帮助。

## 声明

本人郑重声明：所呈交的硕士学位论文《SVG 技术在 WebGIS 中的应用研究》，是本人在导师夏斌的指导下，独立进行研究工作所取得的成果。除文中已经引用的内容外，本文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式表明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名　吴彬卓

2005 年 6 月 5 日