

青海师范大学学位论文独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其它人已经发表或撰写过的研究成果，也不包含为获得青海师范大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中做了明确的说明并表示了谢意。

研究生签名：

日期：

青海师范大学学位论文使用授权声明

青海师范大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其它复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权由青海师范大学研究生部办理。

研究生签名：

导师签名：

日期：

基于 GIS 的气象灾害风险评估系统的设计与实现

中文摘要

我国天气气候复杂,干旱、洪涝、台风、高温、沙尘暴等气象灾害频繁发生,对人民的生命财产造成危害,随着全球气候变暖,极端天气事件加剧,气象灾害已经严重阻碍了我国经济的发展,影响和谐社会的构建。因此掌握气象灾害的特点和发生规律,开展气象灾害风险评估研判,对于预防灾害发生和提高防灾减灾能力有着非常重要的作用,对于保障人民生命财产安全,保证社会经济健康发展有着十分重要的意义。

本文在前人的研究基础上,结合自身的专业技能,以业务需求为出发点,采用 CSharp 编程语言设计开发了基于 GIS 的气象灾害风评估系统。完成的主要工作如下:

- (1) 文章中精细解析了 ESRI 的 ShapeFile 数据文件、ERDAS 的 IMG 数据文件、新一代天气雷达数据文件,并在此基础上定义了新的数据文件 Wr 数据文件、Wv 数据文件、Pyr 数据文件,并实现了对各种数据文件的读写操作和数据文件的相互转换工作;
- (2) 本系统实现了 GIS 中经典的空间分析算法,并把这些算法成功移植到 Wr 文件和 Wv 文件格式中,自定义的 Pyr 文件实现了栅格数据的快速显示操作;基于气象灾害的成灾理论和研判方法,建立了气象灾害研判模型,并在此基础上基于自定义文件设计和实现了气象灾害风险研判系统;
- (3) 系统中用 CSharp 语言编写了 Model.dll 和 Common.dll 两个类库,其中 Model.dll 类库中完成了自定义文件结构的设计和系统中使用的基本实体的定义,Common.dll 类库中实现了对自定义文件的基本操作和经典的 GIS 空间分析算法以及数据格式间的相互转换;
- (4) 设计开发了气象灾害风险调查图制作模块和风险灾害区划图制作模块,系统在风险自动识别研判的基础上设计开发了专家二次研判和专家二次绘制研判功能。根据专家经验对自动识别研判不满意之处可以重新设置各个研判等级的范围重新研判,还可以手动制定研判等级绘制区域,使得系统的研判误差更小,研判准确性更高,系统不经可以把各种调查图和灾害风险区划图以及研判结果保存为数据文件,还可以导出为专题图,还可以导出指定的风险等级内的承载体信息,并对这些信息做分类统计,根据需要通报给相关部门。

关键词: 气象灾害, 风险研判, CSharp, 自定义数据文件, GIS

Abstract

Climatic of China complex, chilling injury, high temperature, droughts, floods, typhoons, sand storms and other weather disasters are frequent, be harm to the life and property of the people, as the global warming, extreme weather events are very frequent. It is very important significance of disaster prevention and control that knowing Characteristics and occurrence of meteorological isaster.

In this article on the basis of previous research, combine myself expertise to business needs in mind, the CSharp programming language designed to develop the risk analysis system of meteorological disaster based on the custom file. Major work completed are as follows:

(1) Article analysed the ESRI ShapeFile data files, ERDAS IMG files, the RadNex data file, defined Wv、Pyr、Wr data files on the basis of those data files, and enabled read and write operations on a variety of data files and convert data files;

(2) This system has completed space analysis algorithm, and put these algorithm successly transplantation to Wr file and Wv file format, Pyr file has gated lattice data of fast displayed operation;

(3) System has completed Model.dll and Common.dll base on Csharp Language. Model.dll defined some base objects and Common.dll defined some operations.

(4) This system designed and developmented of meteorological disaster risk mapping module and disaster risk zoning map of the production module, system designed and developmented the second judging base on the first judging.

Key words: Meteorological hazards, risk analysis, CSharp, custom file, GIS

目录

摘要	I
Abstract	II
第一章 引言	1
1.1 研究背景.....	1
1.2 研究现状及进展.....	1
1.3 研究目的及意义.....	3
1.4 研究内容.....	3
第二章 气象灾害风险成灾模型及研判方法	5
2.1 气象灾害风险成灾模型.....	5
2.2 气象风险研判方法.....	7
第三章 文件解析及数据文件格式自定义	9
3.1 ESRI 的矢量文件格式解析	9
3.2 ERDAS 的 IMG 文件格式解析	12
3.3 新一代天气雷达 (NEXRAD) 数据格式解析.....	16
3.4 数据文件格式自定义.....	19
第四章 基于 GIS 的气象灾害风险评估系统的设计与实现	23
4.1 基于 GIS 的气象灾害风险评估系统设计.....	23
4.2 基于 GIS 的气象灾害风险评估系统实现.....	30
第五章 系统应用及效果评价	57
5.1 系统应用.....	57
5.2 效果评价.....	62
第六章 结论与展望	64
6.1 结论.....	64
6.2 展望.....	65
第七章 参考文献	66
致 谢	68

第一章 引言

1.1 研究背景

我国是世界上受自然灾害影响最严重的国家之一。近些年来,中国每年因自然灾害造成的经济损失占 GDP 的 3%~6%,重灾年份损失可超过 6%^[2],我国从 20 实际中期以来,许多年份都出现旱涝和台风等多种气象灾害。过去的自然灾害研究侧重于灾害的自然因素分析,如灾害分布规律、形成机理、趋势预测等,对风险评估研究较少。自然灾害又包括地质灾害和气象灾害,同一种气象灾害连年连季出现,如 1951 年-1980 年我国华北地区出现春夏连旱和伏秋连旱的年份有 14 次。近年来,我国的地质灾害评估理论已经比较成熟,但气象灾害风险评估却处在萌芽阶段,且气象灾害的防灾减灾、重大基础设施规划和建设等亟需风险评估和区划成果作为科技支撑,因此,有必要对其进行系统探讨。

我国天气气候复杂,干旱、洪涝、台风、高温、冷害、沙尘暴等气象灾害频繁发生,气象灾害造成的损失严重,受气象灾害影响的人口达 4 亿多人。台风每年登录我国平均 8 个,台风深入内陆可造成华北等地的暴雨,暴雨山洪还会引发山体崩塌滑坡、泥石流等灾害,每年造成经济损失高达几十亿元。同时由气象灾害引发或衍生的其它灾害,都对国家经济建设、人民生命财产安全构成极大威胁,光铁道部门每年用于政治险阻工程费用多大 10 亿元。同时有科学依据证明,今后数十年有些气象灾害的发生频率和强度有可能增加^[3]。随着全球气候变暖,极端天气现象在我国频繁发生,气象灾害已经严重阻碍我国经济的发展。近年来,专家学者从遥感监测、气象学特征、时空分布等方面对各种气象灾害进行了大量的研究,其中灾害风险区划是灾害研究领域的一个重要方面,并取得了丰硕的成果^[4-7]。研究表明,掌握气象灾害的特点和发生规律,开展相应的气象灾害风险评估工作,对于防御气象灾害,提高防灾减灾能力,趋利避害,保障人民生命财产安全,保证社会经济健康发展具有十分重要的意义。

1.2 研究现状及进展

目前气象灾害风险的系统评估还没有一个成熟的普遍实用的理论模型,大多数研究是充分借鉴比较成熟的地质灾害风险评估的理论与方法。不同类型的气象灾害有着不同的孕灾环境,因此不同类型气象灾害的风险因素不完全相同,且不同的风险因素在风险评估中的作用大小也是不同的。在构建孕灾环境、危险性、易损性和防灾减灾能力评估模型的时

候, 由于统计数据的限制或者资料的欠缺, 有些因素很难量化, 导致指标因子不全面。不易量化的气象灾害风险因素在评估中如何体现, 还有待进一步研究。在实际应用中, 灾害风险区划所涉及的行政区域越小越好, 如果行政区划精确到乡镇乃至村, 那么区划结果应用价值就会更高。

大约在 20 世纪 70 年代后, 气象灾害评价工作正式兴起^[8], 到今天, 出现了多种气象风险评价与区划方法, 例如自然灾害理论中的灾害风险评价指数法、统计理论中的概率和统计方法、人间生存环境风险评价方法(LEC 评价法)、综合评价法(如层次分析法、灰色关联度分析法等)、基于信息扩散理论的风险评价方法、基于 GIS 的风险评价与区划方法、基于气象一作物产量分析的作物灾害风险分析与评价方法、模糊聚类风险区划方法等。国内外研究人员针对各种方法进行了相应的研究, 并得出很多重要的成果。

Blaikei 提出灾害是承灾体脆弱性与致灾因子综合作用的结果^[9]; Petak W 提出了自然灾害风险评价方法^[10]; BURTONIR、MILETIDS、PETAKWJ^[11]等从灾害系统角度对评价体系进行探讨, 得出很多重要的结论。我国气象灾害风险评估始于 20 世纪 50 年代, 主要以洪涝和干旱等灾种为研究对象, 90 年代以后, 包括洪水、干旱、低温冷害以及台风等在内的气象灾害评估也开始受到重视^[29-32]。盛绍学^[33]建立了安徽省的农业气象灾害定量指标和评估方法、魏瑞江^[34]建立了河北省农业气象灾害定量指标和评估方法、姜爱军^[35]建立了江苏省的农业气象灾害定量指标和评估方法。丁燕等以广东省 14 个市辖区为例, 分别从台风暴雨和台风大风的角度分析了台风致灾因子的时、空、强规律, 并选择人口密度、人均 GDP 和农业占 GDP 的比重等 3 个指标综合反映研究区台风灾害的潜在损失风险^[1]。乐肯堂^[36]对我国海洋的风暴潮灾害进行了风险评估, 并结合相应的灾情资料进行统计分析; 冯定原^[37]等应用灰色系统理论关联分析的原理和方法对新疆伊犁地区的历年畜牧气象灾害进行了分析。谭宗琨对影响广西粮食产量的主要气象灾害进行评估研究, 并对灾害风险进行区划; 陈怀亮^[38]等人根据农业气象灾害风险分析理论, 运用多因子综合风险指数模型, 对河南省小麦生产农业气象灾害风险进行了综合区划, 为农业气象灾害风险探讨了一条新的定量分析及区划技术体系; 刘静^[39]等运用产量逐级分离模拟和逐步筛选聚类分析方法建立了综合灾害等级查询表和灾损评估模型。

随着气象灾害风险评估理论的逐渐发展与完善, 国内相关学者开始利用编程语言, 开发各种灾害风险区划系统。易高流^[40]编制了雷击损害风险评估软件, 为雷击损害风险评估业务提供了一种新的方法和思路; 张斌^[41]等利用 GIS 技术开发了浙江省台风灾害评估系统, 可为全省的台风决策服务提供基础支持; 樊敏剑^[42]利用 3S 技术建立洪水预警和灾害评估系统; 吴浩云^[43]等人利用计算机和 GIS 技术开发了集空间数据库管理、非空间数据库管理、洪涝灾害评估模型库管理以及洪涝灾害预测、评估、输出于一体的太湖流域洪涝灾害评估系统, 为太湖地区防汛抗洪、抢险救灾提供科学依据; 刘俊^[44]等人从江苏省洪涝灾害防治的实际需求出发, 建立了江苏省洪灾害评估系统; 陈建^[45]等编制了贵州省旱情监测与灾害评估系统。

综上可知, 系统对单一灾种进行风险区划的较多, 不可扩展; 风险区划的分辨率低, 没有精细化评估产品; 不同的系统侧重考虑的因子不同, 没有综合考虑; 模型理论多, 实际应用系统少。

1.3 研究目的及意义

我国是一个地域辽阔、人口众多的发展中大国, 人与自然、人与社会怎样和谐相处的矛盾日益突出。在各类自然灾害中, 70%以上是气象灾害, 如何切实加强应急管理, 提高预防和处置我国重大气象灾害的应急能力, 是构建社会主义和谐社会的重要内容, 也是全面履行政府职能、提高行政能力的迫切要求^[12]。据统计, 在所有的自然灾害中, 气象灾害造成的经济损失最大^[13]。随着全球变暖趋势的进一步加剧, 洪涝、风暴、暴雨等水文气象灾害日益频繁, 严重影响世界经济和社会的可持续发展。每次水文气象灾害的发生, 都给上海这座人口密集、经济发达的国际大都市造成重大经济损失, 严重制约了工农业生产发展, 威胁着人民生命财产安全^[14]。气象灾害是影响最大、范围最广的自然灾害, 我国是自然灾害影响严重的国家之一^[15]。在最近几十年里, 由于全球气候变暖, 各地极端天气气候事件和气象灾害发生频繁, 国内外许多专家、学者相继开展了气象灾害风险区划和评价研究等方面的工作^[16]。气象灾害不仅对直接承灾体产生影响, 还通过直接承灾体对其它相关的承灾体产生间接影响, 通常以扩散方式传导, 气象灾害风险呈现出“系统型”的特点^[17]。综合考虑气象条件、社会经济和自然因素, 进行气象灾害风险评估, 能为区域防灾减灾规划提供理论依据^[18]。自然灾害一直伴随着人类社会的生产、生活与发展, 气象灾害尤甚, 它们给人类社会造成的经济损失十分巨大。根据中国气象局资料统计, 在各类自然灾害中, 气象灾害大约占到 70%以上。我国每年重大气象灾害影响的人口大约达 4 亿人次, 所造成的经济损失非常严重。近半个世纪以来, 发生在我国的重大气象灾害, 受灾人口常高达数亿人次, 造成的直接经济损失高达数千亿元^[19]。

随着 GIS 和 RS 技术与现代网络通信技术的日益结合, 其在气象灾害监测、预警和评估中的应用反映出独特的优越性, 可以进行不同灾害的各个要素的监测分析, 从而提高灾害监测、预警和评估的实时性、准确性^[20]。

综上所述, 实现一个多种气象灾害风险研判系统是迫切需要的, 本文是基于这样的初衷和理念, 并设计开发了基于自定义文件的气象灾害风险研判系统。

1.4 研究内容

灾害评估是开展气象灾害防灾减灾评估工作的重要组成部分, 其不仅是在防灾减灾工作对气象部门提出的要求, 更是安全气象理念的体现, 中国气象局有关部门正在积极推动此业务的发展^[21]。本文在前人理论研究和应用的基础之上, 以实际的业务需求为出发点,

以 CSharp 为开发语言，自定义了系统支持的数据文件 Wv、Wr 和 Pyr 数据文件并设计开发了基于自定义文件的气象灾害风险研判系统。本文的主要的研究内容和成果有：

- ①解析了 ESRI 的 Shapfile 数据文件、ERDAS 的 IMG 数据文件、新一代雷达数据文件，并在此基础上定义了新的数据文件 Wr 数据文件、Wv 数据文件、Pyr 数据文件，并且实现了各种数据文件的读写操作和数据文件间的相互转换。
- ②实现了 GIS 中的经典算法，并把这些算法成功移植到了 Wr 文件和 Wv 文件格式中，自定义的 Pyr 文件提高了栅格数据实时显示的效率。
- ③总结当前气象灾害风险研判现状，基于气象灾害成灾理论和研判方法，建立了气象灾害研判模型。
- ④用 CSharp 语言编写了 Model 和 Common 类库。
- ⑤开发了可以在 Windows XP/Windows7 操作系统下稳定运行的基于 GIS 的气象灾害风险评估系统，并设计开发了气象灾害风险调查图制作模块和风险灾害区划图制作模块。

第二章 气象灾害风险成灾模型及研判方法

2.1 气象灾害风险成灾模型

气象灾害风险是指某一地区在某一时期由于气象因子异常或者承灾体承载能力太弱而引起的一系列对人类的生存和社会经济的发展及生态环境的稳定造成损害的可能性^[28]。社会经济影响评估是灾害性天气应急服务的重要环节，是启动灾害应急预案，拟定减灾对策的定量依据。社会经济影响评估主要包括灾前评估和灾后评估两种。所谓灾前评估，也就是气象灾害风险评估。其主要任务是灾前根据可能发生的气象灾害，结合灾区的经济密度、人口密度和抗灾能力等，在灾前预先估测气象灾害的损失，指导启动响应等级的防灾预案，科学合理地开展防灾工作。而灾后评估则是对受灾范围、人口伤亡程度、生产和生活的破坏程度以及潜在的经济损失和环境损失等进行统计、综合和分析，一方面为救灾工作提供科学依据，另一方面是为今后防灾和减灾措施的制定提供科学基础。风险研判就是对选取的致灾因子、孕灾环境、承灾体进行权重分配，以风险评估模型为专业运算基础，绘制灾害风险分布图，研判得到各级抗灾能力承灾体的详细信息，并能自动筛选出预警信息接收对象。

自然灾害的三个主要因子为孕灾环境，致灾因子和承灾体，因而从防灾减灾的角度看，灾情评估应围绕这三方面进行。致灾因子、孕灾环境、承灾体和防灾减灾能力四个因子是相互独立、相互平行的关系，因此气象灾害风险就可以表示为^[1]：

$$\text{RESULT} = U^{w_u} \times M^{w_m} \times N^{w_n} \times [b + (1-b) \times (1-P)] \quad (2.1)$$

$$U = \sum w_{uk} D_{uk} \quad (2.2)$$

$$M = \sum w_{mk} D_{mk} \quad (2.3)$$

$$N = \sum w_{nk} D_{nk} \quad (2.4)$$

$$P = \sum w_{pk} D_{pk} \quad (2.5)$$

式中，RESULT 是气象灾害风险指数，P、N、M、U 分别标识防灾减灾能力、承载体脆弱性、孕灾环境敏感性、致灾因子危险性。 w_u 、 w_m 、 w_n 、 w_p 分别表示对应因子的权重，范围是 0~1， D_k 是指标 k 量化后的值。权重之和等于 1，b 为常数，表示风险灾害的不可防御部分。气象灾害形成的机理可以用下图 2.1 表示



图 2.1 气象灾害形成机理图

（1）致灾因子

由灾害学观点可知，所谓致灾因子是指一切可能引起人员伤亡、财产损失及资源破坏的各种自然与人文异变因素，它是各种灾害、事故发生的危险源。气象致灾因子就是能造成上述灾难性后果的各种气象影响因素^[22]。张继权等认为致灾因子气象灾害危险性(如暴雨、干旱、洪涝等)，它是指可能造成财产损失、人员伤亡、资源与环境破坏、社会系统混乱等孕灾环境的变异因子^[23]，史培军特别指出：人类既是承载体又是致灾因子^[24]。对于一个气象灾害的致灾因子是根据式 2.2 对致灾因子无量纲化后的数值。根据标准，我国气象灾害因子主要有酸雨、雪灾、冷害、大风、雨涝、雷电、干旱、沙尘暴、高温、寒潮、低温、霜冻、冰雹、雾霾等。然而不同的因子，其强度的度量标准不同。一次降水一般按照降水量分为不同的等级。台风以最大风速、最低气压等参数来描述起强度。本文以专家经验把各个不同因子划分为不同的等级。

（2）孕灾环境

孕灾环境是孕育灾害的自然环境，主要包括地理环境、大气环境、天气系统、地形地貌等^[25]。标定区域孕灾环境的定量指标是孕灾环境的稳定程度，孕灾环境的危险度与孕灾环境的稳定性的意义基本相同，指孕灾环境下某段时间内发生气象灾害的频度和程度。气象灾害的发生不仅与致灾因子有关，而且与研究区域所处的自然地质地理环境有关。自然地质环境包括地形地势、地质地貌、河网水系、植被覆盖情况和海拔高度等，孕灾环境可以抑制或促进气象灾害，如同样的降雨量在不同坡度地带导致泥石流的概率不一样。气象灾害系统的强度、灾害系统的群聚和群发特征、灾害等级以及复杂程度是由孕灾环境决定的。对于某一气象灾害孕灾环境因子是根据(式 2.3)的原理进行无量纲化后的数值。综合考虑，按照所占权重叠加处理作为孕灾环境敏感性指标。

（3）承灾体

白美兰认为：承灾体是致灾因子作用的对象，即蒙受灾害的灾体^[25]。史培军指出：

承灾体是包括人类本身在内的物质文化环境，主要有农田、森林、草场、道路、居民点、城镇、工厂等人类活动的财富集聚体^[19]。总结上面的观点可知，可能受到气象灾害直接和间接作用的一切对象都是承灾体。

对于气象灾害承灾体是根据(式 2.4)无量纲化后的数值。一个地区的社会经济发展水平决定了承灾体的易损性。电力、农业密集程度、人员情况、房屋质量、经济发展水平等基础设施情况在一定程度上可以反映该地区承灾体的脆弱性程度。如，干旱的评价指标可选耕地面积占土地面积比重、人口密度、地均 GDP。地均 GDP、耕地面积占土地面积比重、人口密度、农房质量可作为台风和暴雨洪涝灾害的承灾体脆弱性指标。大风的主要评价指标是农房的质量。雾以人口密度、地均 GDP 两项因子作为主要评价指标等。

(4) 防灾减灾能力

防灾减灾能力是人类社会用来应对气象灾害所采取的方针、政策、技术和行动的总称，表示人们对灾害的积极防御程度^[1]。气象灾害能够产生多大风险的重要影响因素就是人类社会的防灾减灾的能力。防灾减灾能力越强，承灾体的脆弱性越弱，灾害风险就越小。防灾减灾能力也是根据(2.5)式原理进行无量纲化后的数值。影响防灾减灾能力的因子是多方面的，总的来说，由工程性防灾减灾措施和非工程性防灾减灾措施。气象灾害系统理论体系如图 2.2 所示



图 2.2 气象灾害理论体系框架层次图

2.2 气象风险研判方法

随着我国经济的快速发展和社会的长足进步，常规的气象灾害评估精度根据研究尺度的不同，从县区级别到省市级别不同，县区内共用一个评估结果，这样的精度已经不能满足政府和人民日益增长的需求，如何实现气象灾害风险的评估，更客观真实的表现研究区内差异是目前气象灾害风险评估的瓶颈之一^[1]。本系统根据气象灾害风险研判原理，分别把致灾因子、承灾体、孕灾环境和防灾减灾能力量化加权进行相应的运算，最终得到研判

结果，并可生成研判专题图。

首先，承灾体的选择方面，本文主要选择研究区域的经济数据、房屋数据、水利设施、林业数据、灾害敏感单位、重大危险源等作为承灾体并把它量化，转化成 Wv 文件保存起来作为系统的数据源。在量化的时候，承灾体的相关信息作为研究区域不同地区的属性数据存储起来，这些属性包括区域 GDP、工业生产总值、农业生产总值、农业人口数、农业家庭数、钻混结构面积、土木结构面积、农村房屋面积、城镇房屋面积、灌溉面积、增库容量等。灾害敏感单位包括学校、医院和企业，其中学校、医院和企业的相关信息也被保存在属性信息里面以及一些其他承灾体都以相同的方式数字化。

其次，致灾因子方面，本文主要选择精细化预报和监测实况作为致灾因子，并可以加载雷达站的雷达图作为辅助研判。精细化预报数据可以按照不同日期不同时间间隔从气象部门直接读取数据，并通过空间插值算法生成 Wr 文件保存起来作为本系统的数据源，供后面研判使用，精细化预报主要有预报降雨量、QPF、QPE 和预报温度；监测实况资料也可以按照不同日期不同时间直接从气象部门获取数据，然后利用空间插值算法保存成 Wr 文件作为系统的数据源，供后面研判使用，监测实况主要包括监测温度、相对湿度、风速、时间段内的平均气温、积温和监测降雨量等。

再次，孕灾环境方面，孕灾环境包括灾害风险区划和灾害易发调查图。其中灾害风险区划根据相关部门的数据，运用空间插值生成相应的风险灾害区划图，插值的过程其实就是量化的过程，结果保存为 Wr 文件，做为后面研判的数据源。灾害易发调查图是工作人员实地调查并记录成文件，然后制作成 Wv 文件，危险等级做为属性存储在 Wv 文件中，作为后面研判的数据源。

最后，防灾减灾能力方面，根据各个地区的防灾减灾设施情况，由专家分别给出防灾减灾能力等级。

第三章 文件解析及数据文件格式自定义

3.1 ESRI 的矢量文件格式解析

Shape files 是 ESRI 提供了一种矢量数据格式，它没有拓扑信息，一个 Shape files 由一组文件组成，其中必要的基本文件包括坐标文件(.shp)、索引文件(.shx)和属性文件(.dbf)三个文件。

3.1.1 坐标文件(.shp)结构

坐标文件用于记录空间坐标信息。它由头文件和实体信息两部分构成。实体信息又包括记录头和记录内容。记录头包括记录号和坐标记录长度。记录内容包括几何类型和具体坐标记录，根据几何类型不同，记录内容的结构有所不同。坐标文件的总体结构如图 3.1 所示。

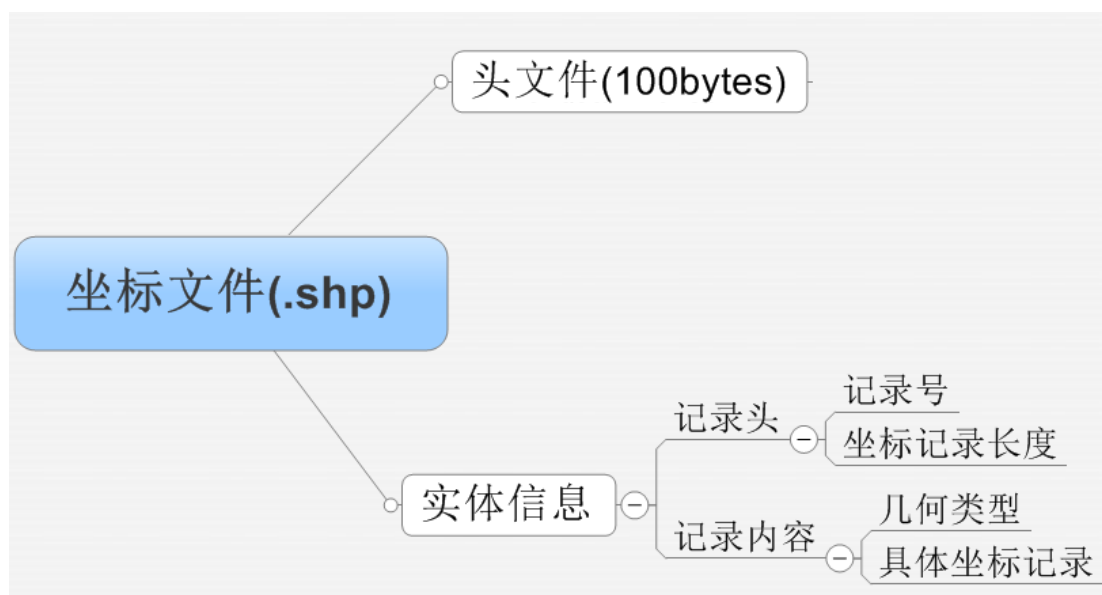


图 3.1 坐标文件(.shp)的总体结构

其中头文件是由 100 个字节组成，坐标文件头文件的详细结构如表 3.2 所示：

表 3.2 坐标文件头文件的详细结构

位置	名称	数值	类型	位序
0	File Code	9994	Integer	big
4	Unused	0	Integer	big
8	Unused	0	Integer	big
12	Unused	0	Integer	big
16	Unused	0	Integer	big
20	Unused	0	Integer	big

24	文件长度	文件的实际长度	Integer	big
28	版本号	1000	Integer	Little
32	几何类型	表示 shape 文件记录的空间数据的几何类型	Integer	Little
36	Xmin	空间数据所占空间范围的 X 方向最小值	Double	Little
44	Ymin	空间数据所占空间范围的 Y 方向最小值	Double	Little
52	Xmax	空间数据所占空间范围的 X 方向最大值	Double	Little
60	Ymax	空间数据所占空间范围的 Y 方向最大值	Double	Little
68	Zmin	空间数据所占空间范围的 Z 方向最小值	Double	Little
76	Zmax	空间数据所占空间范围的 Z 方向最大值	Double	Little
84	Mmin	最小 Measure 值	Double	Little
92	Mmax	最大 Measure 值	Double	Little

记录头由记录号和坐标记录长度组成，其中记录号和坐标记录长度都是 Integer 型，都是 big 位序。

记录内容的结构根据 Shape file 记录的空间数据的几何类型不同而不同，Shape file 支持的空间数据类型如表 3.3 所示

表 3.3 Shape file 支持的空间数据类型

编号	几何类型
0	Null Shape (Shapefile 文件不含坐标)
1	Point (Shapefile 文件记录是点，但不是多点)
3	PolyLine (Shapefile 文件记录是线)
5	Polygon (Shapefile 文件记录是面)
8	MultiPoint (Shapefile 文件记录是多点，即点集合)
11	PointZ (Shapefile 文件记录是三维点)
13	PolyLineZ (Shapefile 文件记录是三维线)
15	PolygonZ (Shapefile 文件记录是三维面)
18	MultiPointZ (Shapefile 文件记录是三维点集合)
21	PointM (含有 Measure 值的点)
23	PolyLineM (含有 Measure 值的线)
25	PolygonM (含有 Measure 值的面)
28	MultiPointM (含有 Measure 值的多点)
31	MultiPatch (表示复合目标)

下面分别介绍点状目标 (Point) (表 3.4 所示)、线状目标 (PolyLine) (表 3.5 所示) 和面状目标 (Polygon) (表 3.6 所示) 三种几何类型的 Shapefile 文件的记录内容。

表 3.4 Point 类型记录内容详细结构

记录项	数值	数据类型	长度	个数	位序
几何类型	1 (表示点)	int	4	1	Little
X 坐标	X 方向坐标值	double	8	1	Little
Y 坐标	Y 方向坐标值	double	8	1	Little

表 3.5 PolyLine 类型记录内容详细结构

记录项	数值	数据类型	长度	个数	位序
几何类型	3 (表示线)	int	4	1	Little
坐标范围	当前线目标的坐	double	32	4	Little

	标范围				
子线段个数	当前线目标的子 线段个数	int	4	1	Little
坐标点数	构成当前线目标 所包含的坐标点 个数	int	4	1	Little
Parts 数组	每个子线段的坐 标在 Points 数组 中的起始位置	int	4*子线段个数	子线段个数	Little
Points 数组	记录了所有坐标 信息	Point 型	根据点个数来 确定	坐标点数	Little

表 3.6 Polygon 类型记录内容详细结构

记录项	数值	数据类型	长度	个数	位序
几何类型	5(表示面)	int	4	1	Little
坐标范围	当前面目标的坐 标范围	double	32	4	Little
子线段个数	当前面目标的子 环个数	int	4	1	Little
坐标点数	构成当前面目标 所包含的坐标点 个数	int	4	1	Little
Parts 数组	每个子环的坐标 在 Points 数组中 的起始位置	int	4*子线段个数	子线段个数	Little
Points 数组	记录了所有坐标 信息	Point 型	根据点个数来 确定	坐标点数	Little

3.1.2 属性文件(.dbf)结构

属性文件(.dbf)用于记录属性信息。它是一个标准的 DBF 文件，也是由头文件和实体信息两部分构成。其中头文件的长度是不定长的，主要是对这个 dbf 文件的记录项信息进行了详细地描述，属性文件的头文件的详细结构如表 3.7 所示。

表 3.7 属性文件(.dbf)头文件的详细结构

位置	长度	说明
0	1 byte	表示当前的版本信息
1-3	3 bytes	表示最近的更新日期，按照 YYMMDD 格式
4-7	int	文件中的记录条数
8-9	short	文件头中的字节数
10-11	short	一条记录中的字节长度
12-13	2 bytes	保留字节，用于以后添加新的说明性信息，这里用 0 来填写
14	1 bytes	表示未完成的操作
15	1 bytes	dBASE IV 编密码标记

16-27	12 bytes	保留字节，用于多用户处理时使用
28	1 bytes	DBF 文件的 MDX 标识。在创建一个 DBF 表时，如果使用了 MDX 格式的索引文件，那么 DBF 表的表头中的这个字节就自动被设置了一个标志，当你下次试图重新打开这个 DBF 表的时候，数据引擎会自动识别这个标志，如果此标志为真，则数据引擎将试图打开相应的 MDX 文件
29	1 bytes	Language driver ID
30-31	2 bytes	保留字节，用于以后添加新的说明性信息，这里用 0 来填写
32-X	(n*32) bytes	记录项信息描述数组。n 表示记录项的个数。这个数组的结构在表 3.8 中有详细的解释
X+1	1byte	作为记录项终止标识

表 3.8 记录项信息描述

位置	长度	说明
0-10	11 bytes	记录项名称，是 ASCII 码值
11	1 byte	记录项的数据类型，是 ASCII 码值。(B、C、D、G、L、M、N)
12-15	4 bytes	保留字节，用于以后添加新的说明性信息时使用，这里用 0 来填写
16	1 byte	记录项长度，二进制型
17	1 byte	记录项的精度，二进制型
18-19	2 byte	保留字节，用于以后添加新的说明性信息时使用，这里用 0 来填写
20	1 byte	工作区 ID
21-30	10 bytes	保留字节，用于以后添加新的说明性信息时使用，这里用 0 来填写
31	1 byte	MDX 标识。如果存在一个 MDX 格式的索引文件，那么这个记录项为真，否则为空

其中记录项数据类型 B(二进制)表示各种字符、C(字符型)表示各种字符、D(日期型)用于区分年、月、日的数字和一个字符，内部存储按照 YYYYMMDD、G(General or OLE)各种字符、N 表示数值型、L 表示逻辑型、M 表示各种字符。

属性文件的实体信息就是一条条属性记录，每条记录都是由若干个记录项构成，因此只要依次循环读取每条记录就可以了，属性记录的顺序和 Shape 里记录的顺序保持一致。

3.1.3 索引文件(.shx)结构

索引文件(.shx)主要包含坐标文件的索引信息，文件中每个记录包含对应的坐标文件记录距离坐标文件的文件头的偏移量。通过索引文件可以方便地在坐标文件中定位到指定目标的坐标信息。

索引文件也是由头文件和实体信息两部分构成，其中文件头部分是一个长度固定(100bytes)的记录段，其内容和坐标文件的文件头一样。它的实体信息以记录为基本单位，每一条记录包括偏移量和记录段长度两个记录项，它们的位序都是 big，两个记录项都是 int 型。

3.2 ERDAS 的 IMG 文件格式解析

Img 格式是一种组织结构比较好的数据格式，在各种 GIS 平台的图像处理系统中应用很多^[26]。IMG 文件格式是一种可存储多种类型数据、应用广泛的图像数据格式，Img 存储

信息的重要特点是分块存储，可以根据需要分块读取，只读取需要的块信息，大大的提高了读取速度。

3.2.1 Img 文件格式的基本对象

Img 文件格式是由一系列相互联系的节点构成(如图 3.9 所示)，每个节点的头文件里分别记录着下一个节点、上一个节点、第一个子节点以及父节点的存储位置，可以根据这些信息轻松的读取到所有的节点，而且每个节点的头文件里还存储着各个节点数据的存储位置、数据大小、节点的结构类型等信息，可以根据节点的结构类型轻松的读取每一个节点。每一个节点的结构类型都是不同的，这些结构类型列表存储在文件末尾。

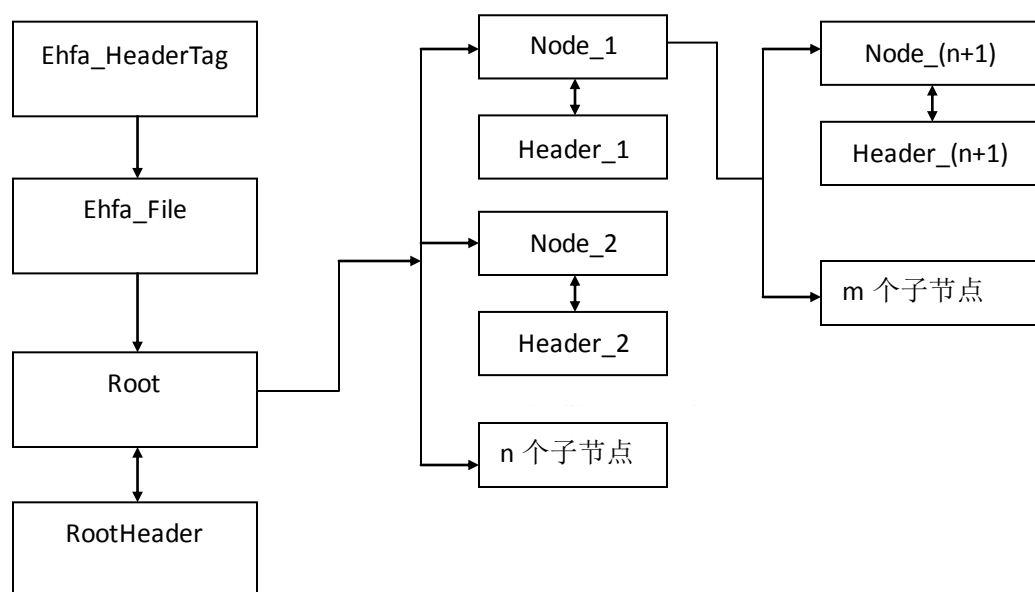


图 3.9 Img 文件结构示意图

其中 Ehfa_HeaderTag 是整个 Img 文件的头文件。Root 是根节点。每一个节点的头文件都按照 Ehfa_Entry 的格式组织。Ehfa_HeaderTag、Ehfa_Entry 和 Ehfa_File 没有自己的头文件，它们不是节点，结构如下：

(1) Img 格式的总文件头 Ehfa_HeaderTag

```
char[16 ] label; //Img 文件的标志 EHFA_HEADER_TAG
int headerPtr; //Ehfa_File 对象的存储位置
```

Ehfa_HeaderTag 存储在整个文件的开头，首先读取这个对象，根据此对象的 headerPtr 信息，读取 Ehfa_File 对象。

(2) Ehfa_File 对象

```
int version; //通常是 1
int freeList; //保留
int rootEntryPtr; //根节点（Root）的存储位置，文件真正信息的开始
short entryHeaderLength; //每一个节点头文件的长度
```

int dictionaryPtr; //各个节点的结构列表的存储位置

Ehfa_File 对象中的 rootEntryPtr 是一个很重要的信息,根据它可以找到根节点 Root 的存储位置,从而开始对文件的真正的读取。

(3) 每个节点的头文件对象 Ehfa_Entry, 是各个节点联系的纽带

int next; //下一个节点的位置

int prev; //前一个节点的位置

int parent; //父节点的位置

int child; //第一个子节点的位置

int data; //此节点数据的存放位置

int dataSize; //数据的大小

char[64] name; //程序员可以自定义的节点的名字

char[32] type; //节点的存储结构的名字

TIME modTime; //此节点的修改时间

此对象是各个节点的头文件格式,前四个变量就是联系各个节点的纽带,通过它们可以轻松的读出 Img 文件中所有的节点。

3.2.2 Img 文件中节点的存储结构

通过读取 Ehfa_File 对象中的 dictionaryPtr 和整个文件流的长度,可以获得 Img 文件中用到的所有的节点结构列表。部分结构如下:

```
{1:lversion, 1:LfreeList, 1:LrootEntryPtr, 1:sentryHeaderLength, 1:LdictionaryPtr, } Ehfa_File,
{1:Lnext, 1:Lprev, 1:Lparent, 1:Lchild, 1:Ldata, 1:ldataSize, 64:cname, 32:ctype, 1:tmodTime, } Ehfa_Entry,
{16:clabel, 1:LheaderPtr, } Ehfa_HeaderTag,
{1:lwidth, 1:lheight, 1:e3:thematic, athematic, fft of real-valued data, layerType,
1:e13:u1, u2, u4, u8, s8, u16, s16, u32, s32, f32, f64, c64, c128, pixelType, 1:lblockWidth, 1:lblockHeight, } Eimg_Layer,
{1:lnumvirtualblocks, 1:lnumobjectsperblock, 1:lnextobjectnum, 1:e2:no compression, RLC compression, compressionType, 0:poEdms_VirtualBlockInfo, blockinfo, 0:poEdms_FreeIDList, freelist, 1:tmodTime, } Edms_State, .....
```

大括号外面最后是节点存储结构的名字,大括号内部是该节点的变量,两个变量之间用逗号好隔开,逗号之间是变量的数量、变量的类型和变量的名称。以下结构的解

释都是基于 C Sharp 语言的。如 1:1width 表示 1 个 int 型的变量 width，此节点的存储结构的名称是 Eimg_Layer，又如 0:poEdms_FreeIDList，这个代表此位置存储着两个 int 型变量，一个表示是 Edms_FreeIDList 的存储位置，另一个表示 Edms_FreeIDList 的个数，再如 64:cname，表示一个 char 数组，数组中有 64 个元素，数组名称为 name。

3.2.3 Img 文件格式中的数据节点

Eimg_Layer 节点是 Img 图像格式中一个最基本的节点结构，该节点存储一个图层（对于多波段影响而言，一般为单个波段）的基本信息，在它的子节点中存储该图层的影像灰度、地理信息、投影信息、统计信息等，Eimg_Layer 的结构如下：

```
int width; //图层的宽度
int height; //图层的长度
Enum layerType; /*图层的类型，是一个枚举，其实这里存储着一个 short 类型，值代表枚举量中的序号，一共三个值，序号分别为 0、1、2{thematic, athematic, fft of real-valued data} */
Enum pixelType; /*图层像素的存储类型，是一个枚举，这里存储着一个 short 类型，值代表枚举量中的序号，一共 13 个值，分别对应 {u1, u2, u4, u8, s8, u16, s16, u32, s32, f32, f64, c64, c128} */
int blockWidth; //图层块的宽度，因为图层是分块处理，后面有介绍
int blockHeight; //图层块的高度
Edms_State 是 Eimg_Layer 的一个子节点，用来描述一个图像块的基本信息，结构如下：
```

```
int numvirtualblocks; //图层中图像块的数目
int numobjectsperblock; //每个图像块的像素个数
int nextobjectnum; //保留值
Enum compressionType; /*是否压缩，是一个枚举，其实这里是一个 short 类型的值，值代表枚举的编号，枚举为 {no compression, RLC compression} */
Edms_VirtualBlockInfo blockinfo; /*图像块的信息，这里存储着两个 int 型变量，一个是图像块的存储地址，另一个是图像块的个数*/
Edms_FreeIDList freelist; //留用信息
TIME modTime; //修改时间
```

其中 blockinfo 是一系列 Edms_VirtualBlockInfo 列表，有多少块就有多少 Edms_VirtualBlockInfo。根据存储地址和数量，可以全部把这些块的信息读出。

Edms_VirtualBlockInfo 的结构为

```
{1:sfileCode, 1:Loffset, 1:lsize, 1:e2:false, true, logvalid, 1:e2:no
compression,
```

ESRI GRID compression, compressionType, }Edms_VirtualBlockInfo, 其中 offset 是本块数据的存储位置, size 是本块数据的大小, compressionTypes 是本块数据的压缩方式。

3.3 新一代天气雷达(NEXRAD)数据格式解析

我国新一代天气雷达业务组网的建设目标是:在我国东部和中部地区,装备先进的新一代 S 频段和 C 频段多普勒天气雷达系统,组成探测空间相互衔接覆盖的监测网,实时监测提供降水强度、平均径向速度和频谱宽度等信息。对降水,特别是暴雨、热带气旋、强对流等灾害性天气和重要天气系统进行有效的监测和警报。我国业务组网的新一代天气雷达(CINRAD),其主要性能已经专家讨论提出,确定采用 WSR-88D 的指标。

新一代天气雷达(NEXRAD)是由产品信息头块(Message Header Block)(表 3.9)、产品描述块(Product Description Block)(表 3.10)、产品符号表示块(Product Symbology Block)(表 3.11)、图形数字文本块(Graphic Alphanumeric Block)(表 3.12)以及文本列表块(Tabular Alphanumeric Block)(表 3.13)构成。其中前两个块是必须块,后三个块是可选块。任何块都要严格按照每个块的结构存储。每块中的字段值是以 HalfWord 方式存储,均采用高字节在前低字节在后的二进制编码格式(即 BIG-ENDIAN 大字节序)^[27]。

表 3.9 产品信息头块数据结构

HalfWord	字段名字	类型	单位	范围	精度	备注
01	MessageCode	int*2	N/A	[-131, -16]&[0, 211]	N/A	
02	Date	int*2	儒略日	[1, 32767]	1	
03-04	Time	int*4	秒	[0, 86399]	1	
05-06	Length	int*4	N/A	[18, 409856]	1	
07	SourceID	int*2	N/A	[0, 999]	1	
08	DestinationID	int*2	N/A	[0, 999]	1	
09	NumberBlocks	int*2	N/A	[1, 51]	1	

表 3.10 产品描述块数据结构

HalfWord	字段名字	类型	单位	范围	精度	备注
10	BlockDivider	int*2	N/A	-1	N/A	
11-12	Latitude	int*4	度	[-90, 90]	0.001	
13-14	Longitude	int*4	度	[-180, 180]	0.001	
15	Height	int*2	英尺	[-100, 11000]	1	海拔
16	ProductCode	int*2	N/A	[16, 131]&[-131, -16]	N/A	

17	Operation	int*2	N/A	[0, 2]	N/A	
18	Volume	int*2	N/A	[1, 767]	1	
19	SequenceN	int*2	N/A	-13&[0, 32767	1	
20	VolumeScan	int*2	N/A	[1, 80]	1	
21	VolumeScanD	int*2	儒略日	[1, 32767]	1	
22-23	VolumeScanT	int*4	秒	[0, 86399]	1	
24	GenerationD	int*2	儒略日	[1, 32767]	1	
25-26	GenerationT	int*4	秒	[0, 86399]	1	
27-28	不同产品有不同的定义					
29	ElevationN	int*2	N/A	[1, 20]	1	仰角数
30-53	不同产品有不同的定义					
54	NumberOfMaps	int*2	N/A	[0, 17]	1	
54	Version	int*1	N/A	[0, 255]	1	
54	SpotBlank	int*1	N/A	[0, 1]	1	
55-56	OffsetSymbology	int*4	HalfWords	[0, 80000]	1	
57-58	OffsetGraphic	int*4	HalfWords	[0, 80000]	1	
59-60	OffsetTabular	int*4	HalfWords	[0, 80000]	1	

其中, 31-46Halfwords 字段是颜色表等的划分, 产品 32、33、81 除外, 其他产品的编码按照如下规则:

如果最 bit 0 是 1, 那么 8-15 位被按照如下编码:

0= “BLANK”

1= “TH”

2= “ND”

3= “RF”

如果 bits 2、3、4、5、6 或 7 是 1, 那么它们将按照如下编码方式编码:

bit 2- 低位字节缩小 20 倍, 保留两位小数精度

Bit 3-低位字节缩小 10 倍, 保留一位小数精度

bit 4= “>”

bit 5= “<”

bit 6= “+”

bit 7= “-”

如果 bit 0 是 0, 那么低位字节表示数值

Example: 数据级别值 (8401, bit 1000 0100 0000 0001) 被解释为<TH

表 3.11 产品符号表示块数据结构

字段名	类型	单位	范围	精度	备注
BlockDivide	int*2	N/A	-1	N/A	块分隔符
BlockID	int*2	N/A	1	N/A	块标识
LengthBlock	int*2	N/A	[1,80000]	1	此块的字节长度

NumberLayers	int*2	N/A	[1,15]	1	块中的层数
LayerDivider	int*2	N/A	-1	N/A	层分隔符
LengthDataLayer	int*4	N/A	[1,80000]	1	显示数据包的字节长度
DisplayDataPackets	N/A	N/A	N/A	N/A	显示数据包

表 3.12 图形数字文本块数据结构

字段名	类型	单位	范围	精度	备注
BlockDivide	int*2	N/A	-1	N/A	块分隔符
BlockID	int*2	N/A	2	N/A	块标识
LengthBlock	int*4	N/A	[1,65535]	1	此块的字节长度
NumberPages	int*2	N/A	[1,48]	1	总也面数
PageNumber	int*2	N/A	[1,48]	N/A	当前页面
LengthPage	int*2	N/A	[4,1360]	1	TextPacker 中的字节数
TextPacket(N)	N/A	N/A	N/A	N/A	显示数据包

表 3.13 文本列表块数据结构

字段名	类型	单位	范围	精度	备注
BlockDivide	int*2	N/A	-1	N/A	块分隔符
BlockID	int*2	N/A	3	N/A	块标识
LengthBlock	int*4	N/A	[1,65535]	1	此块的字节长度
这里是又一个产品信息头块					
这里是又一个产品描述块					
BlockDivider	int*2	N/A	-1	N/A	块分隔符
NumberPages	int*2	N/A	[1,48]	1	总页面数
NumberCharActers	int*2	N/A	[0,1360]	1	本页中的字符数
CharcterData	char	8bit ASCII		N/A	
EndPageFlag	int*2	N/A	-1	N/A	页结束标志

新一代雷达数据是严格按照上述结构存储的，读取新一代天气雷达产品数据时只需严格按照上述结构一步一步读取即可，不同的产品有不同的数据包与之相对应。产品与数据包的对应关系如图 3.14 所示

产品名称 [Ⓔ]	产品号 [Ⓔ]	产品标识 [Ⓔ]	分辨率 [Ⓔ] (KM) [Ⓔ]	覆盖范围 [Ⓔ] (极坐标,KM) [Ⓔ] (笛卡儿坐标,KM x KM) [Ⓔ]	仰角 [Ⓔ]
基本反射率(Packet 44831) [Ⓔ]	19 [Ⓔ]	R [Ⓔ]	1.0 [Ⓔ]	230 [Ⓔ]	0.5、1.5、2.4 [Ⓔ]
	20 [Ⓔ]	R [Ⓔ]	2.0 [Ⓔ]	460 [Ⓔ]	0.5、1.5、2.4 [Ⓔ]
基本速度(Packet 44831) [Ⓔ]	26 [Ⓔ]	V [Ⓔ]	0.5 [Ⓔ]	115 [Ⓔ]	0.5、1.5、2.4 [Ⓔ]
	27 [Ⓔ]	V [Ⓔ]	1.0 [Ⓔ]	230 [Ⓔ]	0.5、1.5、2.4 [Ⓔ]
组合反射率(Packet 47623) [Ⓔ]	37 [Ⓔ]	CR [Ⓔ]	1.0x1.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
	38 [Ⓔ]	CR [Ⓔ]	4.0x4.0 [Ⓔ]	460 [Ⓔ]	Ⓔ
回波顶(Packet 47623) [Ⓔ]	41 [Ⓔ]	ET [Ⓔ]	4.0x4.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
VAD 风廓线(Packet 10) [Ⓔ]	48 [Ⓔ]	VWP [Ⓔ]	2.0 m/s [Ⓔ]	N/A [Ⓔ]	Ⓔ
弱回波区(Packet 47623) [Ⓔ]	53 [Ⓔ]	WER [Ⓔ]	1.0 [Ⓔ]	50x50 [Ⓔ]	Ⓔ
风暴相对径向速度(Packet 44831) [Ⓔ]	56 [Ⓔ]	SRM [Ⓔ]	1.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
垂直累积液态水含量 [Ⓔ]	57 [Ⓔ]	VIL [Ⓔ]	4.0x4.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
风暴追踪信息 [Ⓔ]	58 [Ⓔ]	STI [Ⓔ]	N/A [Ⓔ]	345 [Ⓔ]	Ⓔ
中尺度气旋 [Ⓔ]	60 [Ⓔ]	M [Ⓔ]	N/A [Ⓔ]	230 [Ⓔ]	Ⓔ
1小时降水 [Ⓔ]	78 [Ⓔ]	OHP [Ⓔ]	2.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
3小时降水 [Ⓔ]	79 [Ⓔ]	THP [Ⓔ]	2.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
风暴总降水 [Ⓔ]	80 [Ⓔ]	STP [Ⓔ]	2.0 [Ⓔ]	230 [Ⓔ]	Ⓔ
反射率等高面位置显示 (CAPPI) [Ⓔ]	110 [Ⓔ]	CAR [Ⓔ]	1.0 [Ⓔ]	230 [Ⓔ]	Ⓔ

图 3.14 产品与数据包对应图

3.4 数据文件格式自定义

在解析 ESRI 的适量文件和 ERDAS 的 IMG 文件的基础下，本论文自定义了三种数据文件格式，分别是 WV 文件、WR 文件和 Pyramid 文件，分别对应于矢量文件、栅格文件和金字塔文件。由于栅格文件一般数据存储量比较大，一次性全部读入内存不现实，所以采用金字塔技术来提高栅格文件的操作效率。由于通用的数据文件不能对数据进行保密，所以本文自定义了数据文件格式，并对文件格式进行加密，能很好的对地理数据和气象数据进行保密。

3.4.1 Wr 数据文件结构

Wr 数据文件是有文件头和数据部分两部分组成。其中文件头提供了 Wr 数据的基本信息，包括像素最大最小值、地理范围等信息，数据部分存储着具体的数值信息。其中文件头详细结构如表 3.14 所示，文件头一共 100 个字节。

表 3.14 Wr 文件头详细结构

文件头	名称	数据类型	描述
	文件标识	int	标识文件类型 9995（栅格文件）
	宽度	int	图像的宽度，以像素表示
	高度	int	图像的高度，以像素表示

	坐标 1 X	double	图像左上角像素的 X 坐标，实际的地理坐标
	坐标 1 Y	double	图像左上角像素的 Y 坐标，实际的地理坐标
	坐标 2 X	double	图像右下角像素的 X 坐标，实际的地理坐标
	坐标 2 Y	double	图像右下角像素的 Y 坐标，实际的地理坐标
	像素最小值	double	像素的最小值
	像素最大值	double	像素的最大值
	投影名称的字符数	Int	
	投影名字	Char[]	使用投影的名字
	保留字节	byte[]	(36 减去投影名称的字符数) 个字节，保留值，供后期拓展

Wr 文件的数据区存储像素的值，按照先行再列存储，每一个像素值都是 float 型，没有数据的像素以 999999 填充。

3.4.2 Wv 数据文件结构

Wv 文件是由文件头、图形数据区和属性信息区三部分组成，其后缀名为.wv。其中文件头共 50 个字节，存储 wv 文件的基本信息，包括文件标识(9994)、属性信息在文件中的偏移量、文件中的记录数、版本号、几何类型、边界信息等，具体结构如表 3. 15。数据区存储 wv 文件的具体图形信息，包括记录号、当前对象的边界信息、构成当前对象所有的点数和点集合、构成当前对象子对象个数和子对象点在点集中的位置，具体结构如表 3. 16。属性信息区存储了字段个数、一条记录中的字节总数、记录项类型、记录项长度、每条记录属性的字节数组等，具体结构如表 3. 17。

表 3.15 Wv 文件头数据结构表

起始位置	名称	类型	描述
0	FileCode	int	文件标识 9994（矢量文件）
4	dbfOffset	int	属性信息开始位置在文件中的偏移量字节表示
8	NumCounts	int	文件中图形数据的记录数
12	Version	int	版本号
16	ShapeType	short	几何类型枚举 0、1、3、5 分别对应 null、point、polyline、polygon
18	Xmin	double	X 方向上最小值
26	Ymin	double	Y 方向上最小值
34	Xmax	double	X 方向上最大值
42	Ymax	double	Y 方向上最大值

表 3.16 Wv 文件数据区结构表

记录号	int	
当前线或面的边界盒 Xmin	double	
Ymin	double	
Xmax	double	

Ymax	double	
当前线（面）所有点的个数	int	
当前线（面）所有点的集合		点坐标是 float 型
构成当前线（面）的子线段（子面）的个数	int	
记录每条子线段（子面）在点集合中的开始位置	int	

表 3.17 Wv 文件属性数据结构

FieldsCount	int	字段个数
RecordByteNum	short	一条记录中的字节数从 shp 文件中读取的值减去 1 因为每条记录第一个值无用
数组开始 name	byte[11]	字段名称
FieldType	byte	记录项数据类型，ASCII 码值 B 二进制、C 字符型、D 日期型、G 各种字符、L 逻辑型、M 各种字、N 数值型
FieldLength 数组结束	byte	记录项长度
依次存储每条记录	Byte[]	存储每条记录的字节数组

3.4.3 Pry 数据文件结构

Pyr 文件由文件头和层数据组成，Pyr 文件数据存储在一个文件中，分别是 Pyr 的文件头和 Pyr 的层数据文件，后缀名分别为.00r 和.pyr，整体结构图如图 3.18 所示文件头中存储基本信息，包括层头列表，每个层头中又存储块头列表，层数据文件中存储所有层数据，每层数据又分块存储。Pyr 相当于 Img 中的金字塔文件，定义出 Pyr 文件很大程度上提高了影像数据的显示效率和速度。在文件头文件中存储着 Pyr 文件的基本信息，包括原始图像的像素大小，原始图像的像素宽度和高度、原始图像的坐标范围、像素最大最小值、Pyr 分层数、像素位深、颜色表大小、颜色表信息、投影名称信息以层头列表，具体数据结构如表 3.19 所示。在层头中存储着每一层的基本信息，包括层 x 方向上的分辨率、层 y 方向上的分辨率、层像素大小、层数据在数据文件中的偏移量、层中块的高度、层中块的宽度、x 方向上分的块数、y 方向上分的块数、每块的经度范围、每块的纬度范围和块头列表，具体数据结构如表 3.20 所示。块头中存储着每一块的基本信息，包括块在层中的编号、此块数据在数据文件中的偏移量、块代表的地理范围。

表 3.19 Pyr 文件头数据结构

文件头	SrcSize	Long	原始图像的像素大小
	SrcWidth	Int	原始图像的像素宽度
	SrcHeight	Int	原始图像的像素高度
	ImgArea	ImgRect	原始图像的坐标范围
	min	double	像素最小值

	Max	Double	像素最大值
	MidTierNum	Int	金字塔层数
	BitCount	Short	像素位深
	RgbTableSize	Int	颜色表大小如果为 0，则没有颜色表
	RgbTable		颜色表
	PronameCount	Int	投影名字的字节数
	proName	Char[]	投影名字字符数组
	TierHead[MidTierNum]		层头列表

表 3.20 Pyr 中层头数据结构

层头	resolutionX	Double	层 X 方向分辨率
	resoulutionY	Double	层 Y 方向分辨率
	size	Long	层像素大小
	offset	Long	层数据在文件中的偏移量
	blockHeight	Int	层块的高度
	blockWidth	Int	层块的宽度
	LonblockNum	Int	经度方向上分的块数
	LatblockNum	Int	纬度方向上分的块数
	lonUnit	double	每块经度范围
	latUnit	double	每块纬度范围
	BlockHeadList	List<BlaocHead>	块头列表

Pyr 数据文件的定义，提升了 Wr 文件的显示效率和速度，使得系统运行起来更加的流畅。

本系统中也可以直接加载通用的数据文件格式(shp、img 等文件格式)，主要的数据源是自定义的数据文件格式，并且系统中提供了通用文件格式向自定义文件格式转换的方法。对于加密的数据必须使用自定义的数据文件格式。

第四章 基于 GIS 的气象灾害风险评估系统的设计与实现

4.1 基于 GIS 的气象灾害风险评估系统设计

4.1.1 系统设计原则

系统设计的好坏在根本上决定了软件系统的优劣。可以说“差的系统设计必定产生差的软件系统”，但是不能保证“好的系统设计必定产生好的软件系统”。因为在设计之前有需求开发工作，在设计之后还有编码，测试和维护工作，无论哪个环节出了差错，都会把好事搞砸了。所以我们在开发软件的时候，要遵循一下原则：

（1）合适性

系统设计的源头是需求，这是由商业目标决定的。设计出恰好满足客户需求的软件，并且使开发方和客户方获取最大的利益，而不是不惜代价设计出最先进的软件。

评估体系结构好不好的第一个指标就是“合适性”，即体系结构是否符合适合于软件的“功能性需求”和“非功能性需求”。人们一般不会在需求文档中指定软件的体系结构，需求与体系结构之间并没有一一对应的关系，甚至没有明显的对应关系。所以可以充分发挥主观能动性，根据需求的特征，通过推理和归纳的方法设计出合适的体系结构。

对于软件系统而言，能够满足需求的设计方案可能有很多种，究竟该选择哪一种呢？大部分开发人员天生有使用新技术的倾向，而这种倾向对开发产品而言可能是不利的。

（2）结构稳定性

体系结构是系统设计的第一要素，详细设计阶段的工作如用户界面设计，数据库设计，模块设计，数据结构设计等等，都是在体系结构确定之后开展的，而编程和测试是最后面的工作。如果体系结构经常变动，那么建筑在体系结构之上的用户界面、数据库、模块、数据结构等也跟着经常变动，用“树倒猢猻散”来比喻很恰当，这将导致项目发生混乱。当前中国有几句流行的至理名言：“稳定压倒一切”、“发展才是硬道理”。发展的前提条件是稳定，社会如此，开发软件产品也是如此的。所以体系结构一旦设计完成，应当在一定的时间内保持稳定不变，只有这样才能使后续工作顺利开展。

（3）可扩展性

可扩展性是指软件扩展新功能的容易程度。可扩展越好，表示软件适应“变化”的能力越强。由于软件是“软”的，那是否所有的软件必须设计能扩展新功能呢？这要视软件的规模和复杂性而定。

如果软件规模很小，问题很简单，那么扩展功能的确比较容易。要是软件的代码只有 100 行，这时就无所谓“可扩展性”了，你想怎么扩展都可以。如果软件规模很大，问题

很复杂，倘若软件的可扩展性不好，那么该软件就像用卡片造成的房子，抽出或者塞进去一张卡片都有可能使房子倒塌。

现代软件产品通常采用“增量开发模式”开发商不断地推出软件产品的新版本，从而不断地获取增值利润。如果软件的可扩展性比较差的话，每次开发新版本的代价就会很高。虽然开发商抓住了商机，但却由于设计水平差而导致没有赚取多少利润，真是要活活气死。体系结构的稳定性是根据那些稳定不变的需求而设计的，体系结构的可扩展性则是依据那些可变的需求而设计的。从字面上看，稳定性和可扩展性似乎有点矛盾。两者之间存在辩证的关系：如果系统不可扩展的话，那么就没有发展前途，所以不能只关心稳定性而忽视可扩展性；而软件系统“可扩展”的前提条件是“保持结构稳定”，否则软件难以按计划开发出来，稳定性是使系统能够持续发展的基础。所以稳定性和扩展性都是体系结构设计的要素。

分层开发是一种重要的体系结构，有着良好的可扩展性，而且在扩充或修改功能时，基本不会破坏原有结构的稳定性。可以参看分层开发思想。

（4）可复用性

复用就是指“重复利用已经存在的东西”。复用不是人类懒惰的表现，而是智慧的表现。因为人类总是在继承了前人的成果，不断加以利用、改进或创新后才会进步。复用有利于提高产品的质量、提高生产效率和降低成本。由经验可知，通常在一个新系统中，大部分的内容是成熟的，只有小部分内容是创新的。一般地，可以相信成熟的东西总是比较可靠的（即具有高质量），而大量成熟的工作可以通过复用来快速实现（即具有高生产效率）。勤劳并且聪明的人们应该把大部分的时间用在小比例的创新工作上，而把小部分的时间用在大比例的成熟工作中，这样才能把工作做的又快又好。

（5）面向用户原则

在设计阶段，软件界面设计时应做到“友好、简洁、易用”。人机交互的时候尽量让用户少敲击键盘和点击鼠标，操作步骤越少越好；系统等待时间越短越好。总之，做到越简洁越好，尽量减少操作难度。

（6）实用性

系统数据组织灵活，可以满足不同应用分析的需求。并在此前提下，应尽可能的降低开发成本、使用成本和系统维护成本

（7）安全性

GIS 开发中有很多数据，维护数据的安全性显得非常重要，不允许用户随意更改、删除数据，尤其是避免数据的损坏

4.1.2 系统开发环境

操作系统：Windows7

开发工具：CSharp 语言.NET Framework 平台

其他：Word2010、Excel2010

4.1.3 系统技术路线

基于 GIS 的气象灾害风险评估系统主要是基于 windows 操作系统、CSharp 开发语言、.Net Framework 框架的业务软件。首先解析了 ESRI 的矢量文件结构、ERDAS 的 IMG 文件结构、新一代气象雷达(NEXRAD)文件结构，又自定义了矢量数据文件结构 WV 文件、栅格数据文件结构 WR 文件、金字塔文件结构 Pyramid 文件，在 CSharp 语言和.Net Framework 框架下设计开发了 WR 文件、WV 文件显示控件 ArtMap 控件，用来支撑自定义数据文件的基本操作。系统的总体技术路线如图 4.1 所示：

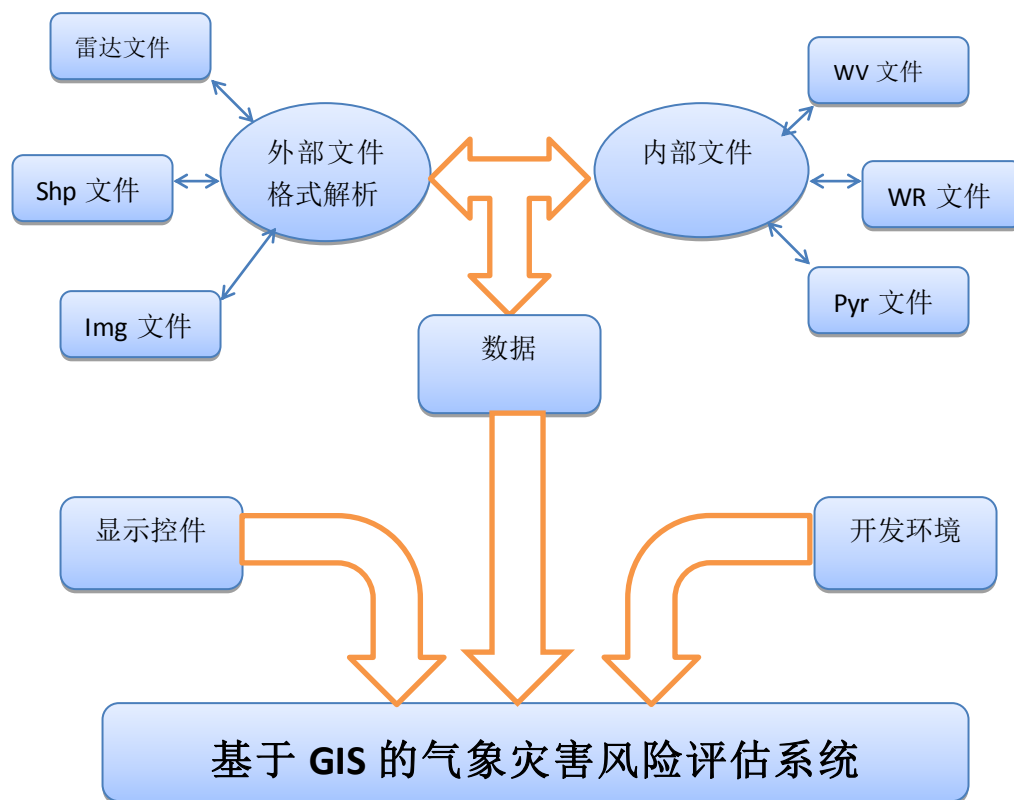


图 4.1 技术路线图

4.1.4 系统开发平台与技术

C# 是一种简洁、类型安全的面向对象的语言，开发人员可以使用它来构建在 .NET Framework 上运行的各种安全、可靠的应用程序。使用 C#，您可以创建传统的 Windows 客户端应用程序、XML Web services、分布式组件、客户端 - 服务器应用程序、数据库应用程序以及很多其他类型的程序。Microsoft Visual C# 2005 提供高级代码编辑器、方便的用户界面设计器、集成调试器和许多其他工具，以在 C# 语言版本 2.0 和 .NET

Framework 的基础上加快应用程序的开发。

CSharp 语言

CSharp 语法表现力强，只有不到 90 个关键字，而且简单易学。C# 的大括号语法使任何熟悉 C、C++ 或 Java 的人都可以立即上手。了解上述任何一种语言的开发人员通常在很短的时间内就可以开始使用 C# 高效地工作。C# 语法简化了 C++ 的诸多复杂性，同时提供了很多强大的功能，例如可为空的值类型、枚举、委托、匿名方法和直接内存访问，这些都是 Java 所不具备的。C# 还支持泛型方法和类型，从而提供了更出色的类型安全和性能。C# 还提供了迭代器，允许集合类的实现者定义自定义的迭代行为，简化了客户端代码对它的使用。作为一种面向对象的语言，C# 支持封装、继承和多态性概念。所有的变量和方法，包括 Main 方法（应用程序的入口点），都封装在类定义中。类可能直接从一个父类继承，但它可以实现任意数量的接口。重写父类中的虚方法的各种方法要求 override 关键字作为一种避免意外重定义的方式。在 C# 中，结构类似于一个轻量类；它是一种堆栈分配的类型，可以实现接口，但不支持继承。除了这些基本的面向对象的原理，C# 还通过几种创新的语言结构加快了软件组件的开发，其中包括：

- (1) 封装的方法签名（称为委托），它实现了类型安全的事件通知。
- (2) 属性（Property），充当私有成员变量的访问器。
- (3) 属性（Attribute），提供关于运行时类型的声明性元数据。

内联 XML 文档注释

在 C# 中，如果需要与其他 Windows 软件（如 COM 对象或本机 Win32 DLL）交互，可以通过一个称为“Interop”的过程来实现。互操作使 C# 程序能够完成本机 C++ 应用程序可以完成的几乎任何任务。在直接内存访问必不可少的情况下，C# 甚至支持指针和“不安全”代码的概念。

C# 的生成过程比 C 和 C++ 简单，比 Java 更为灵活。没有单独的头文件，也不要按照特定顺序声明方法和类型。C# 源文件可以定义任意数量的类、结构、接口和事件。
.NET Framework 平台体系结构。

C# 程序在 .NET Framework 上运行，它是 Windows 的一个必要组件，包括一个称为公共语言运行时（CLR）的虚拟执行系统和一组统一的类库。CLR 是 Microsoft 的公共语言基础结构（CLI）的一个商业实现。CLI 是一种国际标准，是用于创建语言和库在其中无缝协同工作的执行和开发环境的基础。用 C# 编写的源代码被编译为一种符合 CLI 规范的中间语言（IL）。IL 代码与资源（如位图和字符串）一起作为一种称为程序集的可执行文件存储在磁盘上，通常具有的扩展名为 .exe 或 .dll。程序集包含清单，它提供关于程序集的类型、版本、区域性和安全要求等信息。执行 C# 程序时，程序集将加载到 CLR 中，这可能会根据清单中的信息执行不同的操作。然后，如果符合安全要求，CLR 执行实时（JIT）编译以将 IL 代码转换为本机机器指令。CLR 还提供与自动垃圾回收、异常处理和资源管理有关的其他服务。由 CLR 执行的代码有时称为“托管代码”，它与编译为面

向特定系统的本机机器语言的“非托管代码”相对应。

语言互操作性是 .NET Framework 的一个关键功能。因为由 C# 编译器生成的 IL 代码符合公共类型规范 (CTS)，因此从 C# 生成的 IL 代码可以与从 Visual Basic、Visual C++、Visual J# 的 .NET 版本或者其他 20 多种符合 CTS 的语言中的任何一种生成的代码进行交互。单一程序集可能包含用不同 .NET 语言编写的多个模块，并且类型可以相互引用，就像它们是用同一种语言编写的。

除了运行时服务，.NET Framework 还包含一个由 4000 多个类组成的内容详尽的库，这些类被组织为命名空间，为从文件输入和输出到字符串操作、到 XML 分析、到 Windows 窗体控件的所有内容提供多种有用的功能。典型的 C# 应用程序使用 .NET Framework 类库广泛地处理常见的“日常”任务。

4.1.5 系统框架设计

基于自定义文件气象风险研判系统的设计与实现采用框架式设计、构件式开发的方法。将通用的基本功能封装在 common 中，基本的对象封装在 model 中，并编译成动态链接库，通过动态链接库对其调用。业务功能代码实现在业务层面，是对 common 中方法的组合调用，满足业务需求。系统各个层之间通过动态链接库相互调用，修改时只修改动态链接库然后重新调用，这样系统扩展性好，总体框架结构如图 4.2 所示



图 4.2 系统总体框架图

DLL 层是定义的基本结构和一些经典的 GIS 空间分析算法编译后的动态链接库。其中结构定义中包括 Wv、Wr、Pry 等文件结构的定义，还有一点像点线面多点多线多面等基本实体的定义，这些基本实体结构都定义在 Model.DLL 中。在 Common.DLL 中实现了对 shape、img、Wv、Wr、Pry 等文件的读写操作以及转化为图片的操作，空间分析算法也实现在其中。总而言之，在 Model.DLL 中定义了系统所需要的一切实体，Model.DLL 中也指定义实体，不做操作，而在 Commol.DLL 中实现了与实体相关的所有操作以及 GIS 中的经典算法。

业务层主要涵盖了基于自定义文件气象灾害风险研判系统的核心功能，也是完成气象灾害风险评估必不可少的环节。该层各个菜单生成相应的图像数据，各个图像数据构成了影响研判结构的各个因子，这些图像数据是最终研判的依据。研判功能的实现就是对风险灾害区划数据、精细化预报数据、调查数据、天气实况数据等归一化后由专家给出权重叠加得到研判结构，并把研判结果分成不同的等级。

常规服务层用来满足对地图的常规操作，包括了地图放大缩小、全图、平移、地图量算、属性查询、数据管理、地图渲染、涂层编辑、地图出图等基本操作。

数据层包括气象数据、社会经济数据、基础数据。气象数据包括常规气象站观测资料、气象要素派生数据和自动气象观测站资料；社会经济数据包括经济数据、人口数据、基础

设施数据等；基础数据包括交通干线数据、水系数据、DEM 数据、乡镇界数据、村界数据、行政边界数据、自动站点数据、常规站点数据、和网数据等。

设计层主要是对用户体验界面的设计。把地图的基本操作制作成菜单的形式，供用户适用。

4.1.6 系统功能设计

根据气象部门的具体业务需求，基于 GIS 的气象灾害风险评估系统设计了系统模块、地图常规模块、核心功能模块、动态链接库制作模块。主要包括一下功能：致灾因子的数据量化、承灾体的数据量化、孕灾环境的危险度的制作、风险自动识别、专家订正、专家绘制订正、研判结果显示及专题图制作出图、各种属性的查询、研判结果的查询和导出等功能。

①系统模块：为了保证系统能正常运行，需要对系统进行相应的设置来让系统在一个合适的环境中高效的运行。系统模块是系统运行的根本，用以设定系统运行的各种参数。

②常规功能模块：为满足用户日常工作需要，系统提供了地图的一些基本操作、空间分析、专题图、格式转换等功能。地图基本操作实现了常用的地图操作功能，包括放大、缩小、漫游等，还提供地图属性标注、图形要素的选择编辑等功能；属性查询实现了点击地图上一点可以查询该点的基本信息以及研判后的研判信息，点击承灾体可以查询承灾体的基本信息，用户还可以查询感兴趣的目标属性信息和空间信息；空间分析实现了把监测站点检测的实时气象数据转换为地理空间数据(Wr 文件)以及数据格式间的相互转换、栅格计算、渲染着色等。空间分析功能包括格式转换、数据无量纲化、空间差值、投影转换、栅格计算。其中格式转换是把其他数据源的格式转换为自定义的 Wr 或 Wv 文件格式，以备系统兼容，格式转换还包括气象检测站点数据转换为 Wr 文件数据。该操作是系统需要处理多源数据时优先处理的操作；数据无量纲化是进行研判时不同量纲的数据进行空间运算需要无量纲化，处理成同一种数值范围内再进行运算；风险灾害区划数据是各种孕灾环境下的无量纲化后的风险等级数据；空间插值是把各个不同监测站的实时检测数据通过特定的插值方法制作成 Wr 文件；栅格计算在研判时各个研判对象进行栅格计算；投影处理把不同投影信息下的数据处理成同一个投影信息。除上诉功能外，还有用户交互功能。

③核心功能模块包括孕灾环境的风险灾害区划图的制作、调查图制作、各项因子评估、灾害风险研判。其中风险灾害区划图的制作是各种孕灾环境的危险度的量化，有小麦成熟期区划、盛夏绵雨量化、甜橙气候区划图、宽皮桔区划、茶叶区划图、蚕桑区划图、龙眼荔枝区划图、极端高温区划图、暴雨区划图、78 月均温区划图、低端低温分布图、大于 35 度日数分布图、冰雹区划图、春旱区划图、春季低温区划图、地质灾害区划图、冬干区划图、伏旱区划图、高温区划图、雷电区划图、连阴雨区划图、强降温区划图、秋旱区划图、秋季低温区划图、森林火灾区划图、山洪区划图、雾区划图、夏旱区划图；调查图是人为

现场采集信息得到的信息，然后转化为 Wv 文件数据，有地质灾害重点防御区、干旱重点防御区、洪涝重点防御区、森林火险重点防御区、森林火险调查图等；各项因子评定是专家根据经验给孕灾环境、致灾因子、承灾体等赋值权重；灾害风险评估就是实现各个因素赋值权重后的研判运算，得到研判结果和研判数据，并可以查询和导出研判图。

④动态链接库模块包括两部分 Model 和 Common。Model 中定义了系统需要的各种地理实体对象结构，编译后生成 Model.dll 类库。Common 实现了对 Model 定义实体的各种操作和 GIS 中经典的空间分析算法，编译后生成 Common.dll 类库。

4.2 基于 GIS 的气象灾害风险评估系统实现

4.2.1 系统应用环境

（1）硬件配置

中央处理器：四核 2.2GHz 及以上

内存：4GB 以上

硬盘：800G 以上

显卡：独立显卡

外设：打印机等

（2）软件环境

操作系统：Windows、WindowsXP

GIS 环境：ArcMap、ArcCatalog

语言环境：CSharp.NET

其他：Office2007 及以上

4.2.2 气象灾害风险评估系统

基于 GIS 的气象灾害风险评估系统是完全脱离了第三方数据源的自主性很高的系统，通过定义合理的数据文件格式，系统的所有操作都是建立在自定义文件的基础之上的。基于自定义文件的气象灾害风险研判系统实现了客户需求的所有功能。

4.2.2.1 系统主界面

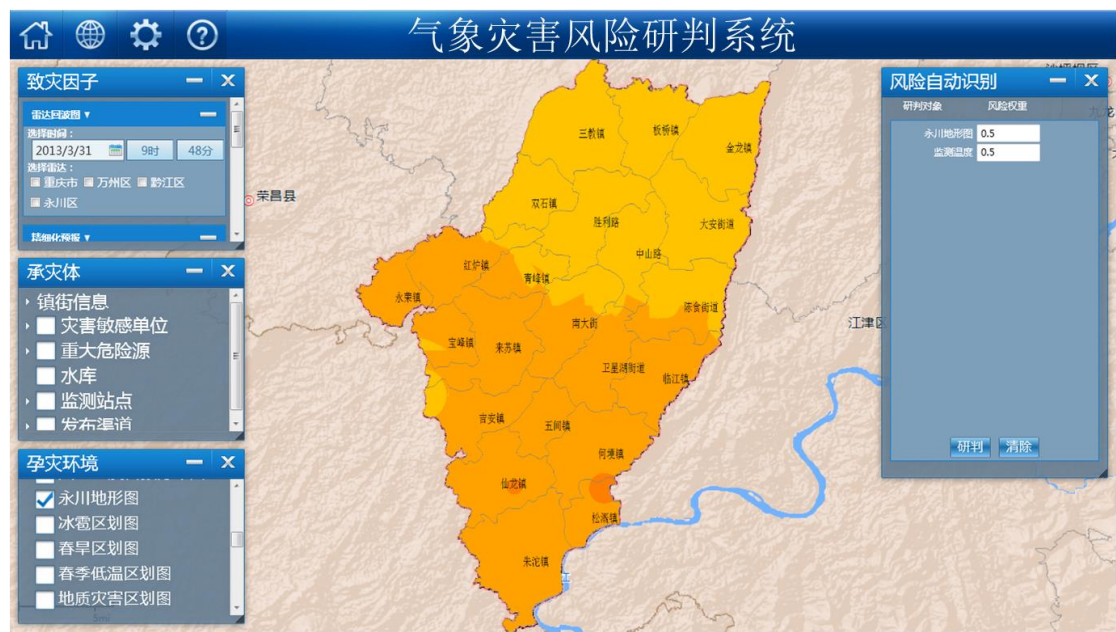


图 4.4 基于 GIS 的气象灾害风险评估系统主界面

程序主界面如图 4.4，分别有系统设置、地图切换、致灾因子、承灾体、孕灾环境、风险自动识别、专家二次研判等主要模块。

系统设置：提供对系统的一些基本设置，有基本设置、服务配置、修改密码等。

致灾因子：主要包括精细化预报、检测实况等。其中精细化预报是不同日期不同时间间隔内的预报降雨量、QPE、QPF、预报温度等。检测实况是不同时间的检测温度、相对湿度、风速等。致灾因子相关内容如图 4.5 所示

精细化预报 ▾

选择时间：2013/4/8

预报间隔：12小时 ▾

选择预报数据：

☐ 预报降雨量 ☐ QPE ☐ QPF

☐ 预报温度

监测实况 ▾

选择时间：2013/4/8 11时 ▾

☐ 监测温度 ☐ 相对湿度

选择实况要素：

☐ 风速

开始时间：2013/4/8 10时 ▾

结束时间：2013/4/8 11时 ▾

☐ 平均气温 ☐ 积温 ☐ 监测降雨量

图 4.5 致灾因子相关信息

承灾体：主要包括研判区经济数据、研判区农业信息、研判区房屋信息、研判区水利设施、研判区林业信息、研判区灾害敏感单位、研判区重大危险源、研判区水库、研判区监测站点。其中经济数据由 GDP、工业生产总值、农业生产总值、人口总数、农业人口数、农业家庭数组成，房屋由钻混结构面积、土木结构面积、农村房屋面积和城镇房屋面积组成，水利设施由灌溉面积和塘库库容量组成，林业主要是森林面积，灾害敏感单位又企业、医院、和学校组成，重大危险源由化危场所防雷防电、地质灾害点、油库加油站、危险品、炸药仓库和其他危险源组成，监测站点由农业监测站点、地质灾害监测站点、水文站、水位监测站点、林场监测站点、气象两要素监测站点、气象四要素监测站点、气象二氧化碳监测站点和气象山洪监测站点组成。承灾体相关内容如图 4.6 所示

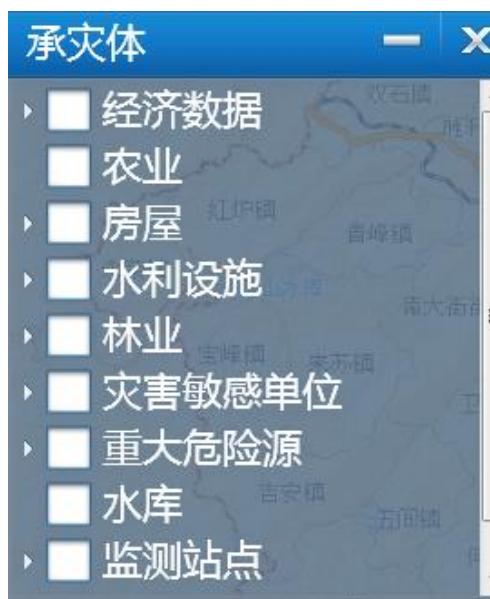


图 4.6 承灾体相关信息

孕灾环境：主要包括灾害风险区划图、灾害易发地区调查图、地形图、土地(如图 4.7)。其中灾害风险区划图的数据源是 Wr 文件，灾害易发地区调查图、地形图、土地的数据源是 Wv 文件。灾害风险区划图中主要有盛夏绵雨、极端高温区划图、暴雨区划图、78 月均温区划图、低端低温分布图、大于 35 度日数分布图、冰雹区划图、春旱区划图、春季低温区划图、地质灾害区划图、冬干区划图、伏旱区划图、高温区划图、雷电区划图、连阴雨区划图、强降温区划图、秋旱区划图、秋季低温区划图、森林火灾区划图、山洪区划图、雾区划图、夏旱区划图(如图 4.8)。灾害易发地区调查图包括地质灾害重点防御区调查图、干旱重点防御区调查图、洪涝重点防御区调查图、森林火险重点防御区调查图、森林火险调查图(如图 4.9)。地形图就是研判区的地形图。



图 4.7 孕灾环境信息图



图 4.8 风险灾害区划信息图

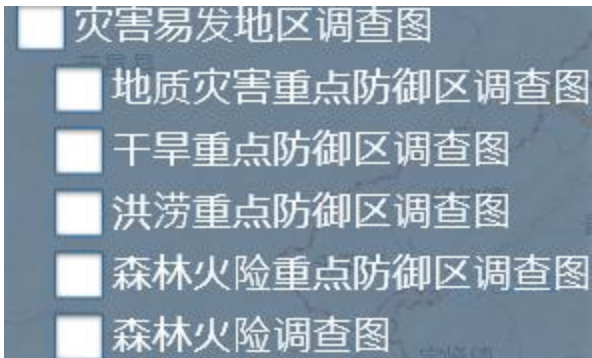


图 4.9 调查信息图

风险自动识别：是把各个研判因子由专家根据经验赋予不同的权重，然后进行自动研判并生成研判结果。

风险专家订正：根据风险自动识别的结果，由专家根据经验对不满意的地方重新设置

权重经行二次研判。

风险专家绘制订正：对研判出来的结果需要手动绘制等级时，由专家手动增加或缩小相应等级的范围。如图 4.10



图 4.10 风险研判功能图

4.2.2.2 系统设置

为了保证系统能正常运行，需要对系统进行相应的设置来让系统在一个合适的环境中高效的运行。系统模块是系统运行的根本，用以设定系统运行的各种参数。系统设置界面如图 4.11



图 4.11 系统设置界面图

4.2.2.3 常规功能实现

地图的放大缩小平移要素选择等基本的地图操作实现是通过定义个工具接口类 (ITool)，然后地图放大 (ZoomIn)、地图缩小 (ZoomOut)、地图全图显示 (ZoomExtent)、地

图漫游 (Pan)、地图要素选择 (Select)、要素属性查询 (Attribute)、地图拉框放大 (BoxZoomIn) 和地图拉框缩小 (BoxZoomOut) 工具分别继承该工具接口类 (ITool)，去实现相应的操作。工具的静态结构模型图如图 4.12 所示。

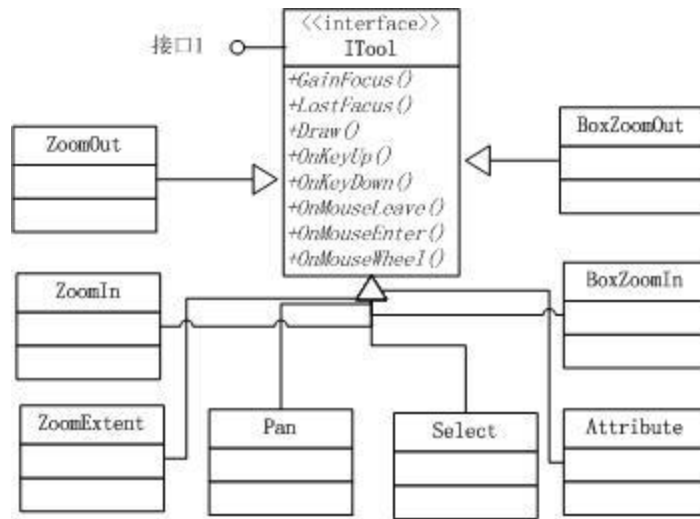


图 4.12 工具静态结构模型图

其中 ITool 类中的主要代码如下：

```

/// <summary>工具接口</summary>
public interface ITool : ICommond
{
    /// <summary>获取焦点</summary>
    void GainFocus();
    /// <summary>失去焦点</summary>
    void LostFocus();
    /// <summary>鼠标按小</summary>
    /// <param name="e"></param>
    void OnMouseDown(MouseEventArgs e);
    /// <summary>
    /// 鼠标拖动</summary>
    /// <param name="e"></param>
    void OnMouseMove(MouseEventArgs e);
    /// <summary>鼠标弹起</summary>
    /// <param name="e"></param>
    void OnMouseUp(MouseEventArgs e);
    /// <summary>鼠标滑轮滚动</summary>
    /// <param name="e"></param>
    void OnMouseWheel(MouseEventArgs e);
}
    
```

```

    /// <summary>鼠标进入控件区</summary>
    /// <param name="e"></param>
    void OnMouseEnter(EventArgs e);
    /// <summary>鼠标离开控件区</summary>
    /// <param name="e"></param>
    void OnMouseLeave(EventArgs e);
    /// <summary>键盘按下</summary>
    /// <param name="e"></param>
    void OnKeyDown(KeyEventArgs e);
    /// <summary>键盘弹起</summary>
    /// <param name="e"></param>
    void OnKeyUp(KeyEventArgs e);
    /// <summary>实时绘制</summary>
    /// <param name="g"></param>
    void Draw(Graphics g);
    /// <summary>工具对应的鼠标样式</summary>
    Cursor Cursor { get; set; }
    /// <summary>地图编辑视图</summary>
    IMapEditView EditView { get; set; }
}

```

各个不同的工具都分别继承 ITool 接口，在各个不同的工具类中实现自己工具相应的操作，以下给出属性查看工具的代码实例。属性查看工具是当鼠标按下时，查看鼠标按下时选择到的要素的属性信息，代码如下：

```

public override void OnMouseDown(System.Windows.Forms.MouseEventArgs e)
{
    _form.Show(); _form.TopMost = true;
    double x, y;
    _disTrans.ToMapPoint(e.X, e.Y, out x, out y);
    IPoint mouseDownPt = new Point(x, y);
    IFeatureLayer featureLayer = _view.GetMapDocument().CurrentLayer
as IFeatureLayer;
    IFeatureClass featureClass = featureLayer.FeatureClass;
    IGeometry geometry = mouseDownPt as Point;
    IQueryFilter queryFilter = new SpatialFilter(geometry, geoSpatialRelationship.GEO_SPATIALRELATIONSHIP_INTERSECTS, null);

```

```

FeatureCursor featureCursor = featureClass.Search(queryFilter)
as FeatureCursor;
Feature feature = featureCursor.Next() as Feature;
IFields fields = feature.Fields;
for (int i = 0; i < fields.Count; i++)
{
    _dataGridView.Rows.Add();
    Field field=fields.GetField(i) as Field;
    _dataGridView.Rows[i].Cells[0].Value = field.Name;
    _dataGridView.Rows[i].Cells[1].Value = feature.GetValue(i)
as string;
}
//绿色显示
IMapDocument _mapDocument = _view.GetMapDocument();
if (_mapDocument.CurrentLayer == null) return;
if (_mapDocument.CurrentLayer is IFeatureLayer)
{
    IFeatureLayer currentLayer = _mapDocument.CurrentLayer as
IFeatureLayer;
    (currentLayer as IFeatureSelection).SelectionSet = curre
ntLayer.FeatureClass.Select(queryFilter);
    _view.Draw();
    _view.Update(_disTrans.WindowBounds);
}
}

```

其它工具的实现和属性查看工具原理一样，只需去实现 ITool 接口里的相应操作即可完成。其他常规功能的实现通过直接实现相应的功能函数来实现。其中格式转换是把其它数据源转换为能为本系统所支持的数据源的一个重要过程，所以把格式转换单独制作成一个工具，来拓宽系统数据源，格式转换工具主要实现了 ESRI 的 shapefile 文件向 Wv 文件的转换以及 ERDAS 的 Img 数据文件向 Wr 数据文件的转换，格式转换界面如图 4.13 所示



图 4.13 格式转换工具界面图

ESRI 的 Shapefile 文件转换为 Wv 文件的核心代码如下：先从 shapefile 文件中读取数据，然后再把数据写入到 Wv 文件中。

读取 Shapefile 文件代码，

```
switch (ShapeType)
{
    case 0://文件为空
        break;
    #region 点的读取
    case 1://Single Point点
        PointF[] pt = new PointF[arr_RecorderNumber]; //定义存放待读点的数组
        polytypestructdata linshil = new polytypestructdata();
        for (int i0 = 0; i0 < arr_RecorderNumber; i0++)
        {
            RecorderNum = bi.ReadInt32(); //记录号
            ContentLength = bi.ReadInt32(); //坐标记录长度
            double x = 0;
            double y = 0;
            shapetype01 = bi.ReadInt32();
            x = bi.ReadDouble(); //X坐标
            y = bi.ReadDouble(); //Y坐标
            linshil.Xmin = Xmin; linshil.Xmax = Xmax;
            linshil.Ymax = Ymax; linshil.Ymin = Ymin;
            linshil.ShapeType = ShapeType;
            linshil.PointNums = arr_RecorderNumber;
        }
    }
}
```

```
linshi1.Arrpoint = new PointF[1];
linshi1.Arrpoint[0].X = (float)x; linshi1
.Arrpoint[0].Y = (float)y;
linshi1.RecorderNum = RecorderNum;
linshi.Add(linshi1);

}
fo.Close();
break;
#endregion
#region 线的读取
case 3://Polyline 线
for (int n1 = 0; n1 < arr_RecorderNumber; n1++)
{
    polytypestructdata linshi2 = new polytypestructdata
    ();
    linshi2.Xmin = Xmin; linshi2.Xmax = Xmax;
    linshi2.Ymax = Ymax; linshi2.Ymin = Ymin;
    linshi2.ShapeType = ShapeType;
    RecorderNum = bi.ReadInt32();//记录号
    linshi2.RecorderNum = RecorderNum;
    ContentLength = bi.ReadInt32();//坐标记录长度
    shapetype01 = bi.ReadInt32();
    double Xmin01 = 0;//边界盒范围
    double Ymin01 = 0;
    double Xmax01 = 0;
    double Ymax01 = 0;
    Xmin01 = bi.ReadDouble();
    Ymin01 = bi.ReadDouble();
    Xmax01 = bi.ReadDouble();
    Ymax01 = bi.ReadDouble();
    int Numparts;//子线段个数
    int Numpoints;//构成当前坐标的坐标点个数
    Numparts = bi.ReadInt32();
    linshi2.Numparts = Numparts;
```

```

        linshi2.Parts = new int[linshi2.Numparts];
        Numpoints = bi.ReadInt32();
        linshi2.PointNums = Numpoints;
        linshi2.Arrpoint = new PointF[linshi2.PointNums];
        PointF[] pt1 = new PointF[Numpoints]; //定义存放待
        读点的数组
        int[] parts = new int[Numparts]; // 记录每个子线段的
        坐标在Points数组中的起始位置
        for (int i0 = 0; i0 < Numparts; i0++)
        {
            parts[i0] = bi.ReadInt32();
        }
        linshi2.Parts = parts;
        double[] pointsX = new double[Numpoints];
        double[] pointsY = new double[Numpoints];
        for (int i0 = 0; i0 < Numpoints; i0++)
        {
            pointsX[i0] = bi.ReadDouble();
            pointsY[i0] = bi.ReadDouble();
            linshi2.Arrpoint[i0].X = (float)pointsX[i0];
            linshi2.Arrpoint[i0].Y = (float)pointsY[i0];
        }
        linshi.Add(linshi2);
    }
    fo.Close();
    break;
#endregion
#region 面的读取
case 5://Polygon 面
    for (int n = 0; n < arr_RecorderNumber; n++)
    {
        polytypestructdata linshi3 = new polytypestructdata
        ();
        linshi3.Xmin = Xmin; linshi3.Xmax = Xmax;
        linshi3.Ymax = Ymax; linshi3.Ymin = Ymin;
    }
}

```

```
linshi3.ShapeType = ShapeType;
RecorderNum = bi.ReadInt32(); //记录号
linshi3.RecorderNum = RecorderNum;
ContentLength = bi.ReadInt32(); //坐标记录长度
shapetype01 = 0; //几何类型
shapetype01 = bi.ReadInt32();
double Xmin01 = 0; //边界盒坐标
double Ymin01 = 0;
double Xmax01 = 0;
double Ymax01 = 0;
Xmin01 = bi.ReadDouble();
Ymin01 = bi.ReadDouble();
Xmax01 = bi.ReadDouble();
Ymax01 = bi.ReadDouble();
int Numparts; //构成当前面状目标的子环个数
int Numpoints; //构成当前面状目标的点数
Numparts = bi.ReadInt32();
Numpoints = bi.ReadInt32();
linshi3.PointNums = Numpoints;
linshi3.Arrpoint = new PointF[linshi3.PointNums];
linshi3.Numparts = Numparts;
linshi3.Parts = new int[linshi3.Numparts];
int[] parts = new int[Numparts]; //记录每个子线段的
坐标在Points数组中的起始位置
for (int i0 = 0; i0 < Numparts; i0++)
{
    parts[i0] = bi.ReadInt32();
}
linshi3.Parts = parts;
double[] pointsX = new double[Numpoints];
double[] pointsY = new double[Numpoints];
PointF[] point1 = new PointF[Numpoints];
for (int i0 = 0; i0 < Numpoints; i0++)
{
    pointsX[i0] = bi.ReadDouble();
```

```

        pointsY[i0] = bi.ReadDouble();
        linshi3.Arrpoint[i0].X = (float)pointsX[i0];
        linshi3.Arrpoint[i0].Y = (float)pointsY[i0];
    }
    linshi.Add(linshi3);
}
fo.Close();
break;
case 8:
    for (int n = 0; n < arr_RecorderNumber; n++)
    {
        polytypestructdata linshi4 = new polytypestructdata
        ();
        linshi4.Xmin = Xmin; linshi4.Xmax = Xmax;
        linshi4.Ymax = Ymax; linshi4.Ymin = Ymin;
        linshi4.ShapeType = ShapeType;
        RecorderNum = bi.ReadInt32(); //记录号
        linshi4.RecorderNum = RecorderNum;
        ContentLength = bi.ReadInt32(); //坐标记录长度
        linshi4.ShapeType = bi.ReadInt32();
        //边界盒
        bi.ReadBytes(32);
        linshi4.PointNums = bi.ReadInt32();
        linshi4.Arrpoint=new PointF[linshi4.PointNums];
        for (int nn1 = 0; nn1 < linshi4.PointNums; nn1++)
        {
            linshi4.Arrpoint[nn1].X = (float)bi.ReadDouble
            ();
            linshi4.Arrpoint[nn1].Y = (float)bi.ReadDouble
            ();
        }
        linshi.Add(linshi4);
    }
    break;
default:

```

```
        break;
    #endregion
}
```

写入 Wv 文件核心代码:

```
#region 写入数据
for (int i = 0; i < NumCounts; i++)
{
    writebi.Write(data[i].RecorderNum);
    writebi.Write(data[i].Xmin); writebi.Write(data[i].Ymin);
    writebi.Write(data[i].Xmax); writebi.Write(data[i].Ymax);
    writebi.Write(data[i].PointNums);
    for (int pt = 0; pt < data[i].PointNums; pt++)
    {
        writebi.Write(data[i].Arrpoint[pt].X); writebi.Write
            (data[i].Arrpoint[pt].Y);
    }
    writebi.Write(data[i].Numparts);
    for (int part = 0; part < data[i].Numparts; part++)
    {
        writebi.Write(data[i].Parts[part]);
    }
}
#endregion

else if (ShapeType == 1)
{
    #region 计算属性信息的偏移量
    offset = 50 + 20 * NumCounts;
    #endregion

    #region 写文件头
    writebi.Write(FileCode); //文件标识 9994标识为Wv文件
    writebi.Write(offset - 1); //属性信息开始位置在整个Wv文件中的偏
    移量
    writebi.Write(NumCounts); //文件中图形记录数
    writebi.Write(Version); //版本号
}
```

```

writebi.Write(ShapeType); //几何类型
writebi.Write(Xmin); //X方向上最小值
writebi.Write(Ymin); //Y方向上最小值
writebi.Write(Xmax); //X方向上最大值
writebi.Write(Ymax); //Y方向上最大值
#endregion
#region 写数据
for (int i = 0; i < NumCounts; i++)
{
    writebi.Write(data[i].RecorderNum);
    for (int pt = 0; pt < 1; pt++)
    {
        writebi.Write(data[i].Arrpoint[pt].X); writebi.Write
            (data[i].Arrpoint[pt].Y);
    }
}
#endregion
}
else if (ShapeType == 8)
{
    #region 计算属性信息的偏移量
    for (int i = 0; i < NumCounts; i++)
    {
        offset += 48 + 16 * data[i].PointNums;
    }
    offset += 50;
    #endregion
    #region 写文件头
    writebi.Write(FileCode); //文件标识
    writebi.Write(offset - 1); //属性偏移量
    writebi.Write(NumCounts); //文件中记录数
    writebi.Write(Version); //版本号
    writebi.Write(ShapeType); //几何类型
    writebi.Write(Xmin); //X方向上最小值
    writebi.Write(Ymin); //Y方向上最小值

```

```

writebi.Write(Xmax); //X方向上最大值
writebi.Write(Ymax); //Y方向上最大值
#endregion
#region 写入数据
for (int i = 0; i < NumCounts; i++)
{
    writebi.Write(data[i].RecorderNum);
    writebi.Write(data[i].PointNums);
    for (int jjj = 0; jjj < data[i].PointNums; jjj++)
    {
        writebi.Write(data[i].Arrpoint[jjj].X); writebi.Write
        (data[i].Arrpoint[jjj].Y);
    }
}
#endregion
}
#region 写入属性信息
writefo.Seek(offset - 1, SeekOrigin.Begin); //按照属性信息偏移量定
位
writebi.Write(FieldsCount); //属性字段个数
writebi.Write(RecordByteNum); //一条记录中属性信息的字节数
for (int i = 0; i < FieldsCount; i++)
{
    writebi.Write(Fields[i].Name); //字段名称
    writebi.Write(Fields[i].FieldType); //记录项类型
    writebi.Write(Fields[i].FieldLength); //记录长度
}
for (int i = 0; i < NumCounts; i++)
{
    writebi.Write(AttrData[i].bytes); //存储每一条记录所有属性的字
    节数组
}
#endregion

```

空间插值是运用数学方法根据观测点数据或样本数据推算区域内其他任一点的值，以便描述和研究连续型球面状实体的空间分布和变化特征的技术，在 GIS 中，空间插值主

要用于估算栅格中每个像元的值，它是将点数据转化为面数据的一种方法。本文采用反距离权重(IDW)插值方法，IDW(Inverse Distance Weighted)是一种常用而简便的空间插值方法，它是基于相似相近的原理，即两个物体里的近，它们的性质就越相似，反之，离得越远则相似性越小，它以插值点与样本点间的距离为权重进行加权平均，离插值点越近的样本点赋予的权重越大。

4.2.2.4 核心功能实现

本文中的空间插值是为了把各个气象监测站的监测数据插值成区域内的类面状数据，制作成灾害风险区划数据并得到灾害风险区划图，插值生成灾害风险区划界面如图 4.14 所示

图 4.14 插值生成灾害风险区划界面图

调查图是通过实地走访权威部门（如森林防火、防汛、地质灾害防治、植保站等）以及有经验的乡镇干部、群众，摸清县域、乡镇、村等某一地域的以下内容：

各类自然灾害（如气象灾害、洪旱灾害、地质灾害）最易发生区、较易发生区、不易发生区等孕灾环境地理分布；

易受灾的承灾体类(如需要纳入预警服务的主要农作物种植区、林区、重要园区、基地)地理分布；

某一指定海拔范围段的地理分布（自动制作）；

虽然绘制出的调查图，其绘制范围是经验性的，精度不是很高，但它在很大程度上反映了一个地区孕灾环境的脆弱性，以及承灾体的分布情况，对于气象防灾减灾风险预警服务而言，具有十分重要的参考价值，有了这些调查图资料，如果收集的其他社会经济基础

数据资料齐全，通过研判平台，在灾害发生前或发生中，可以快速评估出各地的灾害风险等级、某类灾害的各级危险区域，并分析出灾害风险区域内的经济、人口、建筑、农业布局等数据，为风险服务提供神奇而强大的数据支撑，调查图其实有很大部分反映的是地区的防灾减灾能力，这些数据的真实性很强，是风险研判的一个重要指标，所以把调查图的制作放在了核心模块里。

调查图制作工具的主要功能有绘制基于实地调查的孕灾环境分布图、绘制基于实地调查的承灾体分布图、绘制基于实地调查的灾后灾害分布图、绘制与高程数据密切相关的孕灾环境或承灾体分布图。

调查图的制作步骤如下：

组织调查小组开展风险调查：打印出具有乡镇边界的行政区划图，对辖区内各个乡镇的洪涝、干旱、森林火险、地质灾害等重点防御区进行风险调查；

调查中，用笔在打印出的地图上绘制灾害风险区域，绘制的时候要尽量与被调查的人员所描述的范围相符合。

调查结束后，通过调查图制作功能，将图绘制在电子地图上，形成电子文档，存储在Wv文件中。

调查图的制作的操作流程如下：

第一步，模版设置。模版设置包括新建模版、添加风险等级、设置颜色等级。

新增配色模版

配色模版

所属类型： 编号： 名称：

配色模版项

选择	名称	类型	最小值	最大值	颜色	透明值	是否启用
☐	低风险区	雨量调查图	0	50	0070FF	FF	可用
☐	中等风险区	雨量调查图	50	100	FFFF00	FF	可用
☐	高风险区	雨量调查图	101	150	FF0000	FF	可用

首页 上一页 1 of 1 下一页 末页

图 4.15 新建模版界面图

系统中的调查图制作模块可以新增调查图模板，点击详细名称可对模板属性进行修改和查看。

添加模版项

X

配色信息

名称

所属类型

雨量调查图▼

最大值

最小值

透明值

是否可用

☒

颜色

Theme Colors

Standard Colors

More Colors...

新增

取消

图 4.16 添加风险等级并设置等级颜色界面图

调查图模版设置完毕后，开始调查图的制作，模版设置为调查图制作设置好了每个级别的颜色和范围。

第二步，调查图制作。调查图制作包括新建调查图、调查图绘制、保存与导出。

添加调查图

名称

所属类型

雨量调查图

图片地址

文件地址

保存

取消

图 4.17 新建调查图界面

图中所属类型选择第一步设置的模版。



图 4.18 绘制调查图界面



图 4.20 调查图应用到研判系统效果图

调查图制作完毕后可以到处为 Wv 文件格式保存，作为系统研判功能的数据源。

研判功能的实现是在对承载体、孕灾环境、致灾因子等研判因子量化的基础上，调用 Common. dll 里风险研判功能接口，根据气象风险研判理论，对各个图层进行加权叠加，得到研判结果，研判结果图保存在 Wr 文件中。研判功能有风险自动识别研判，专家二次研判和专家绘制研判。专家二次研判和专家绘制研判的设计理念是对自动识别研判不满意时可以根据专家经验手动调整各个等级的比例或绘制指定等级区域然后重新研判。各种研判的结构实现都在 Common. dll 类库中，可以直接调用。研判界面如图 4.21

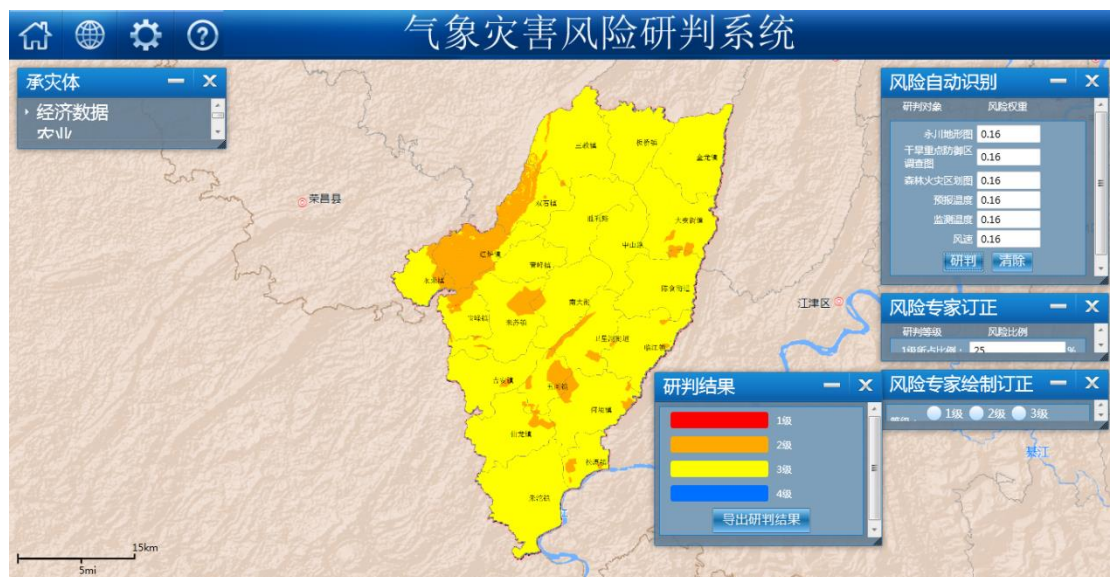


图 4.21 风险研判功能界面

4.2.2.5 动态链接库(DLL)模块实现

DLL 模块由两部分组成，分别是 Model.dll 和 Common.dll。Model.dll 中定义了系统需要的基本实体(Object)结构信息，自定义文件 Wv、Wr、Pry 文件就在 Model.dll 中定义，在 Model.dll 中还重新定义了点、线、面、多点、多线、多面等基本实体结构，本系统中采用的均是重新定义的数据结构。Common.dll 中是针对实体的相应操作，Wv、Wr、Pry 的读取与写入、Shapefile 到 Wv 的转换、Img 到 Wr 的转换都在 Common.dll 中实现。完成实体定义和相关操作实现后，直接编译成 Model.dll 和 Common.dll 两个类库，然后气象灾害风险研判系统引用两个类库，并使用 Model.dll 的实体和 Common.dll 里的方法，这样就很大程度减少了系统中的代码量，如需修改，只需对 Model.dll 和 Common.dll 修改后，重新引用即可，方便快捷。编译后的 dll 如图 4.22

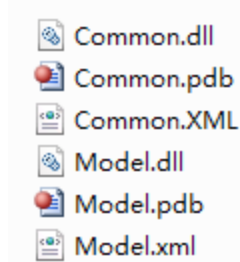


图 4.22 DLL 图

Model 中定义的实体：Wv 文件结构，详细的定义了本系统中使用的 Wv 文件结构，矢量文件结构；Wr 文件结构，详细的定义了本系统使用的 Wr 文件结构，栅格文件结构；Pryamid 文件结构，详细的定义了 Pry 文件结构，金字塔文件结构；Annotation 文件结构，注释文件结构；Bmp 结构，系统中使用的图片结构，是系统显示的关键的结构；PointStyle 结构、LineStyle 结构、PolygonStyle 结构分别是点、线、面符号结构，图层渲染时必不

可少的结构；Color 结构，系统中的颜色结构，RGB 颜色系统；Geometry 中定义了 Point 结构、Line 结构、Polygon 结构，分别是系统中使用到的点线面结构，另外还定义出了 ESRI 的 shapefile 文件结构和 ERDAS 的 IMG 文件结构，因为 shapefile 文件和 img 文件是两个重要的数据源，系统数据很多都是由这两种数据转换而来。Model 编译后生成 Model.dll 类库，供系统直接引用。

Common 中实现的操作：有八大类组成，分别是 API、Common、CoorProjectiong、FileReadWrite、ImgHandle、MathBase、SpatialAnalyst 和 GID。API 中封装了 System 系统的 API，主要提供 DIB 显示函数；Common 中提供字节位序的转换函数、颜色格式的转换函数、文件相对路径与绝对路径的转换函数、数据文件出图函数等；CoorProjection 中实现了投影系统的转换函数；FileReadWrite 中实现了各种文件数据的读与写，包括 Bmp 文件的操作、img 文件的操作、Pry 文件的操作、shapefile 文件的操作、Wr 文件的操作、Wv 文件的操作、Sytle 文件的操作，下面将给出文件读写的关键代码；ImgHandle 中实现了图像的基础操作，包括快速生成位图函数、图片的旋转函数、图像缩放函数、内存法填色函数、数据裁切函数等；MathBase 中实现了基本的数学公式，包括计算经纬度坐标的两点距离函数；判断点与多边形位置关系的函数、计算两条线间的夹角函数、获取平行线函数、判断两线段空间关系函数、计算线段交点函数、计算多边形几何中心函数、×乘积函数、判断点线位置关系函数、多边形面积计算函数、道格拉斯普克-最小线性位移函数、点到直线距离函数等；SpatialAnalyst 中实现了空间分析相关函数，包括创建等值线函数、构建三角格网函数、构建缓冲区函数、空间插值函数、点集抽稀函数等；GDI 中实现了图像绘制相关函数，包括绘点函数、绘线函数、绘面函数等。Common 编译后生成 Common.dll 类库，供系统直接调用。

Wv 文件有数据文件和属性信息两部分组成，下面给出读取 Wv 数据文件和属性信息文件的核心代码：

数据文件代码：

```
ShpDataInfo[] result = new ShpDataInfo[NumCounts];
    if (ShapeType == 0) { }
    else if (ShapeType == 1)
    {
        for (int i = 0; i < NumCounts; i++)
        {
            result[i].ShapeType = ShapeType;
            result[i].RecorderNum = DIYbi.ReadInt32();
            result[i].Arrpoint = new PointF[1];
            result[i].Arrpoint[0] = new PointF();
            result[i].Arrpoint[0].X = DIYbi.ReadSingle();
```

```

        result[i].Arrpoint[0].Y = DIYbi.ReadSingle();
        result[i].Xmin = Xmin; result[i].Ymin = Ymin;
        result[i].Xmax = Xmax; result[i].Ymax = Ymax;
    }
}
else if (ShapeType == 8)
{
    for (int i = 0; i < NumCounts; i++)
    {
        result[i].ShapeType = ShapeType;
        result[i].RecorderNum = DIYbi.ReadInt32();
        result[i].PointNums = DIYbi.ReadInt32();
        for (int il = 0; il < result[i].PointNums; il++)
        {
            result[i].Arrpoint[il] = new PointF();
            result[i].Arrpoint[il].X = DIYbi.ReadSingle();
            result[i].Arrpoint[il].Y = DIYbi.ReadSingle();
        }
        result[i].Xmin = Xmin; result[i].Ymin = Ymin;
        result[i].Xmax = Xmax; result[i].Ymax = Ymax;
    }
}
else if (ShapeType == 5 || ShapeType == 3)
{
    for (int i = 0; i < NumCounts; i++)
    {
        result[i].ShapeType = ShapeType;
        result[i].RecorderNum = DIYbi.ReadInt32();//
        result[i].Xmin = DIYbi.ReadDouble();
        result[i].Ymin = DIYbi.ReadDouble();
        result[i].Xmax = DIYbi.ReadDouble();
        result[i].Ymax = DIYbi.ReadDouble();
        result[i].PointNums = DIYbi.ReadInt32();
        result[i].Arrpoint = new PointF[result[i].PointNums];
        for (int i0 = 0; i0 < result[i].PointNums; i0++)

```

```

        {
            result[i].Arrpoint[i0] = new PointF();
            result[i].Arrpoint[i0].X = DIYbi.ReadSingle();
            result[i].Arrpoint[i0].Y = DIYbi.ReadSingle();
        }
        result[i].Numparts = DIYbi.ReadInt32();
        result[i].Parts = new int[result[i].Numparts];
        for (int j = 0; j < result[i].Numparts; j++)
        {
            result[i].Parts[j] = DIYbi.ReadInt32();
        }
    }
}

```

属性信息文件代码:

```

DIYfo.Seek(dbfOffset, SeekOrigin.Begin);
int FieldCount = DIYbi.ReadInt32();//字段个数
short BytesPerField = DIYbi.ReadInt16();//一条记录中的字节数
Shape.FieldsStr[] FieldsIn = new Shape.FieldsStr[FieldCount];
Shape.Dbffileds[] DbffieldsIn = new Shape.Dbffileds[FieldCount];
;
for (int k = 0; k < FieldCount; k++)
{
    byte[] name = DIYbi.ReadBytes(11);
    FieldsIn[k].name = System.Text.Encoding.Default.GetString(name);
    byte b = DIYbi.ReadByte();
    char b1 = (char)b;
    FieldsIn[k].fieldType = b1;
    byte length = DIYbi.ReadByte();
    FieldsIn[k].FieldLength = (int)length;
    DbffieldsIn[k].Name = name; DbffieldsIn[k].FieldType = b;
    DbffieldsIn[k].FieldLength = length;
}
Shape.Attribute[] Attribution = new Shape.Attribute[NumCounts];
Shape.RecordBytes[] RecordB = new Shape.RecordBytes[NumCounts];
for (int k = 0; k < NumCounts; k++)

```

```

{
    RecordB[k].bytes = DIYbi.ReadBytes((int)BytesPerField);
    Attribution[k].str = new string[FieldCount];
    int n = 0;
    for (int k0 = 0; k0 < FieldCount; k0++)
    {
        byte[] linshi = new byte[FieldsIn[k0].FieldLength];
        for (int k00 = 0; k00 < FieldsIn[k0].FieldLength; k00++)
        {
            linshi[k00] = RecordB[k].bytes[n++];
        }
        string nnnn = System.Text.Encoding.Default.GetString(
            linshi);
        Attribution[k].str[k0] = nnnn;
    }
}

```

Wr 文件是本系统中另一种重要的数据源，所以一下给出 Wr 文件读取的核心代码：

```

Wr.Wrdata result = new Wr.Wrdata();
#region 读取头文件
FileStream readfo = new FileStream(DIYPath, FileMode.Open,
    FileAccess.Read);
BinaryReader bi = new BinaryReader(readfo);
int Sign = bi.ReadInt32();
result.Width = bi.ReadInt32();
result.Heigth = bi.ReadInt32();
result.UpperLeftCenterX = bi.ReadDouble();
result.UpperLeftCenterY = bi.ReadDouble();
result.LowerRightCenterX = bi.ReadDouble();
result.LowerRightCenterY = bi.ReadDouble();
result.MinValue = bi.ReadDouble();
result.MaxValue = bi.ReadDouble();
result.ProNameCount = bi.ReadInt32();
result.ProName = bi.ReadChars(result.ProNameCount);
byte[] yuliu = bi.ReadBytes(36 - result.ProNameCount);
#endregion

```

```
#region 读取数据区
result.DataVale = new float[result.Heigth, result.Width];
for (int i = 0; i < result.Heigth; i++)
{
    for (int j = 0; j < result.Width; j++)
    {
        result.DataVale[i, j] = bi.ReadSingle();
    }
}
#endregion
return result;
```

在 Common 中还实现了 GIS 中的经典算法-空间插值、缓冲区分析等，这里不再一一给出代码，这些算法都是实现本系统的核心代码。

第五章 系统应用及效果评价

5.1 系统应用

基于 GIS 的气象风险灾害评估系统能够根据出现的气象灾害及时的研判出在风险区的承灾体信息，并能提供各种灾害区划图以及研判灾害等级通过各种渠道发布，以便及时公布通知相关部门并做好防范和救护工作，减小气象灾害造成的各种损失。由于本人在重庆相关单位实习的缘故，获取重庆各区县的相关数据比较方便，所以本章节以重庆永川区为例，来对气象灾害进行研判分析。

精细化预报数据：重庆永川各个日期内不同时间间隔(12 小时、24 小时、36 小时、48 小时、60 小时、72 小时、84 小时、96 小时、108 小时、120 小时、132 小时、144 小时、156 小时和 168 小时)的预报降雨量、预报温度以及 QPE 和 QPF 数据。

检测实况数据：重庆永川各个日期一天 24 小时各个时段、各个监测站的检测温度、相对湿度和风速数据，还有不同时间间隔内的平均气温、积温和检测降雨量数据。

雷达回波数据：重庆永川雷达站的雷达数据。

社会经济数据：重庆永川区域土地面积、耕地面积、GDP、工业生产总产值、农业生产总值、人口总数、农业人口数、农业家庭数等(气象局提供此类数据)；房屋数据，砖混结构面积、土木结构面积、农村房屋总面积、城镇房屋总面积等(相关部门提供此类数据)；水利设施，重庆永川灌溉面积、增库库容量(相关部门提供此类数据)；重庆永川林业数据；重庆永川灾害敏感单位，企业、医院、学校；重大危险源，危化场所防雷防电、地质灾害点、油库加油站、危险品、炸药库等；监测站点等。

灾害风险区划数据：永川盛夏绵雨区划图、永川极端高温区划图、永川暴雨区划图、永川 78 月均温区划图、永川低端低温分布图、永川大于 35 度日数分布图、永川冰雹区划图、永川春旱区划图、永川低温区划图、永川地质灾害区划图、永川冬干区划图、永川伏旱区划图、永川高温区划图、永川雷电区划图、永川连阴区划图、永川强降温区划图、永川秋旱区划图、永川秋季低温区划图、永川森林火灾区划图、永川山洪区划图、永川雾区划图和永川夏旱区划图以及地质灾害重点防御区调查图、干旱重点防御去调查图、洪涝重点防御区调查图、森林火险重点防御区调查图。

下面以图片形式说明系统应用效果：承灾体选择灾害敏感单位(企业、医院、学校)，致灾因子选择降雨量，孕灾环境选择春旱区划图和春季低温区划图，经行研判。

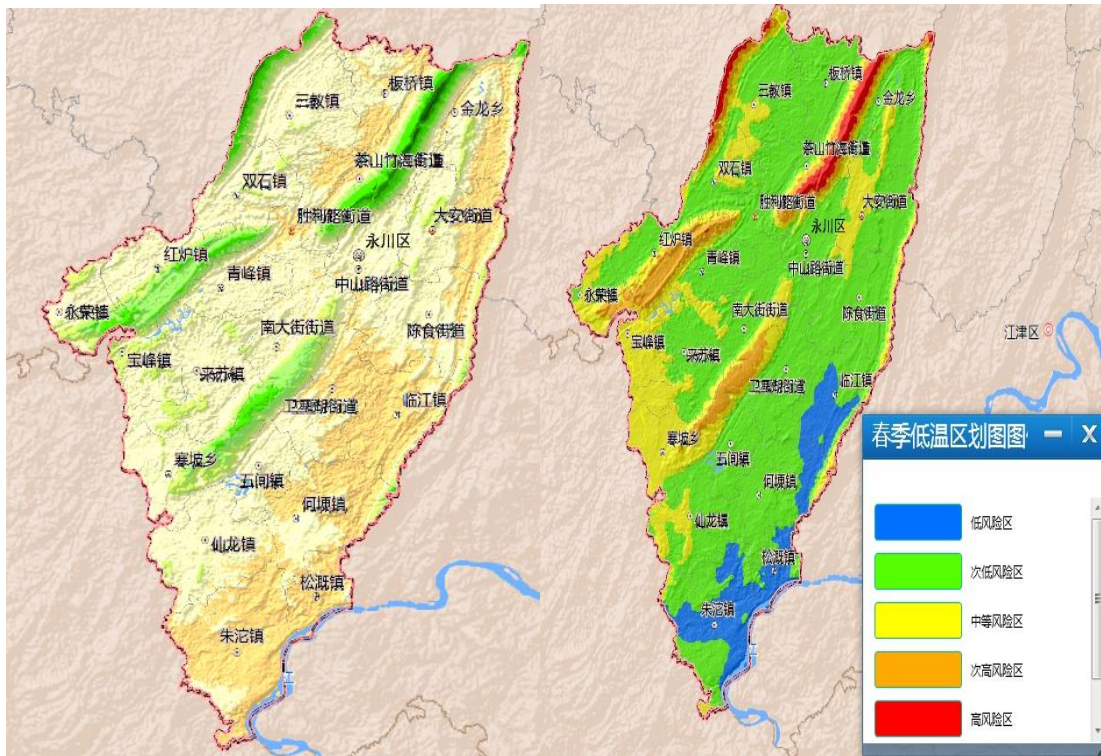


图 5.1 永川地形图

5.2 永川春季低温区划图

首先加载致灾因子，雷达回波图(如图 5.3 所示)、监测实况图(如图 5.4 所示)、精细化预报图(如图 5.5 所示)。雷达回波数据、监测实况数据、精细化预报数据都是实时数据，从气象部门读取的实时数据，插值生成的面状数据（以 W_r 文件格式存储）。

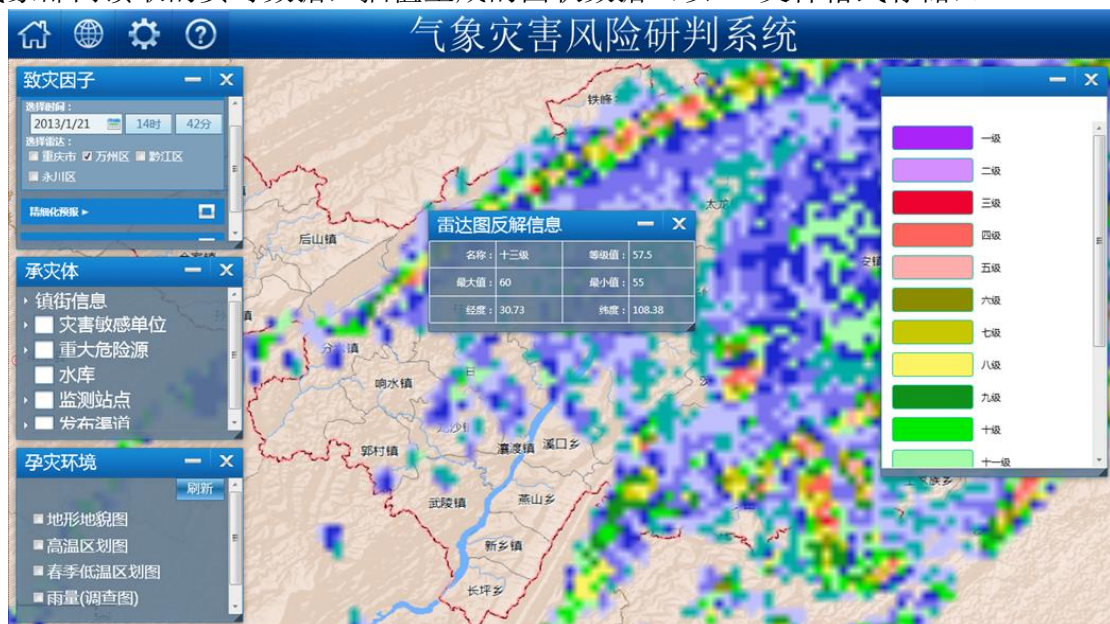


图 5.3 雷达回波图



图 5.4 监测实况图

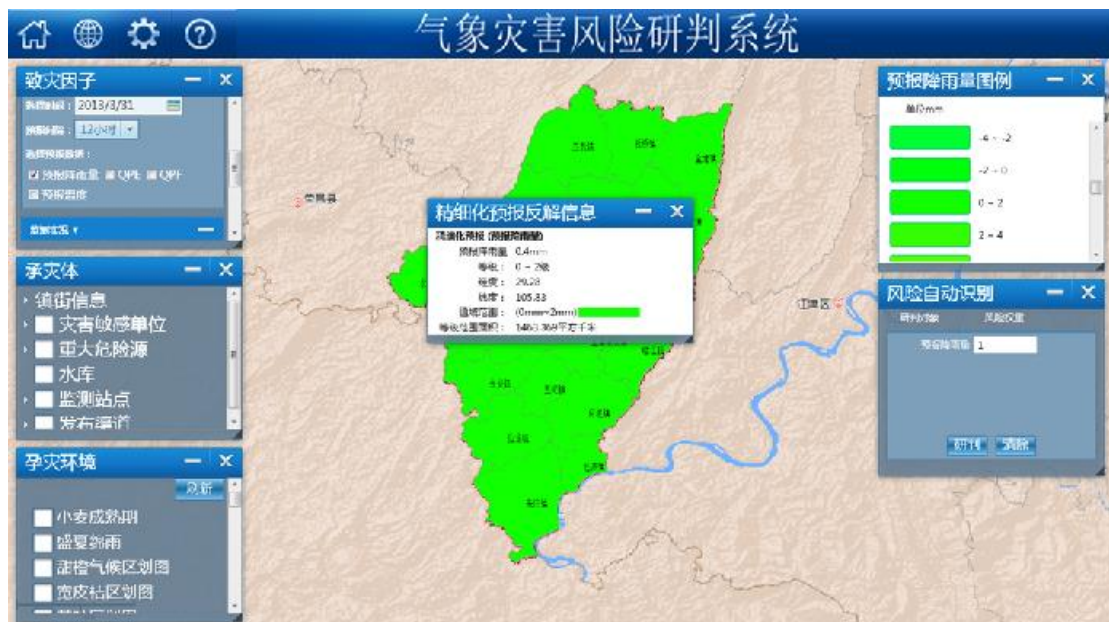


图 5.5 精细化预报图

其次加载干旱区重点防御调查图(如图 5.6)和森林火灾区划图(如图 5.7)。调查图是已经制作好的 W_v 数据，森林火灾区划图也是根据历史数据插值生成的 W_r 文件。

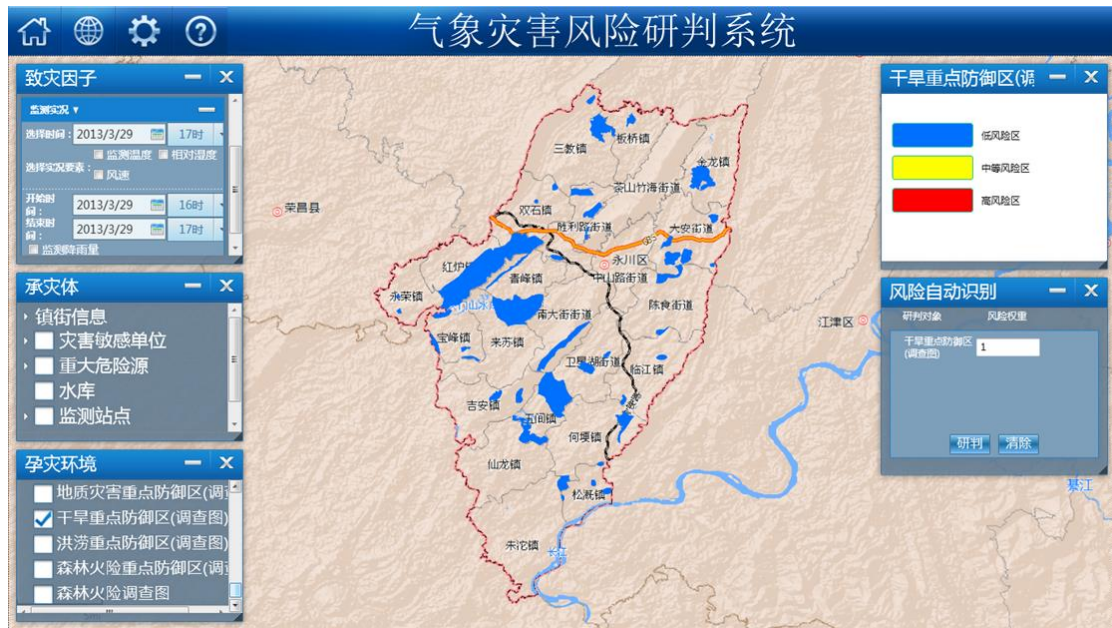


图 5.6 干旱区重点防御调查图



图 5.7 森林火灾区划图

选取致灾因子、承灾体、孕灾环境进行风险研判，至少选取 2 个以上才具有研判意义。如果选取 N 个研判对象，则默认每个对象的研判权重为 n 分之一，可自定义各研判对象的权重。点击研判，结果如图 5.8 所示

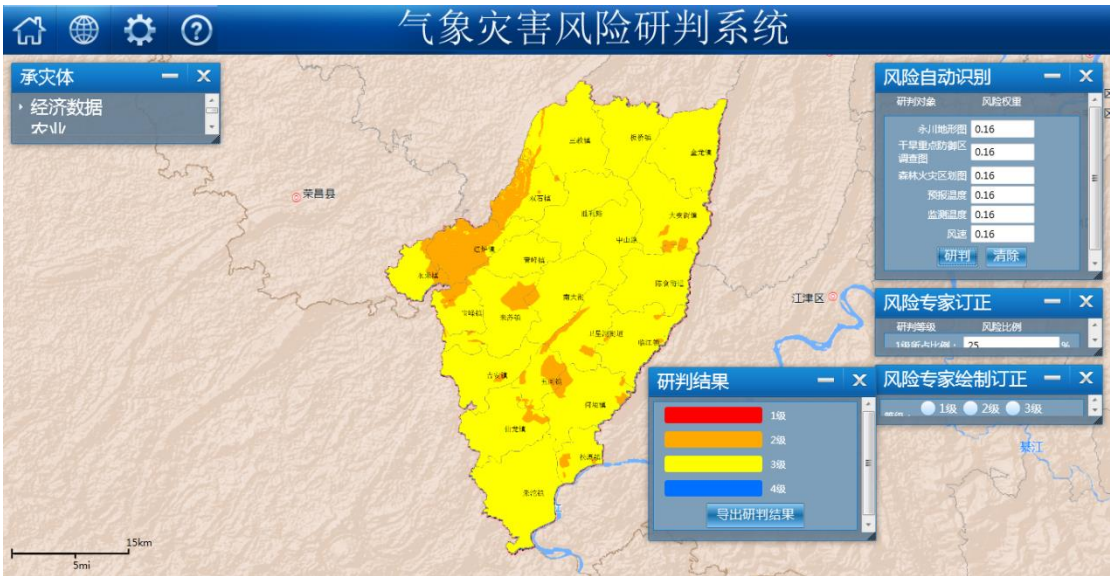


图 5.8 研判结果图

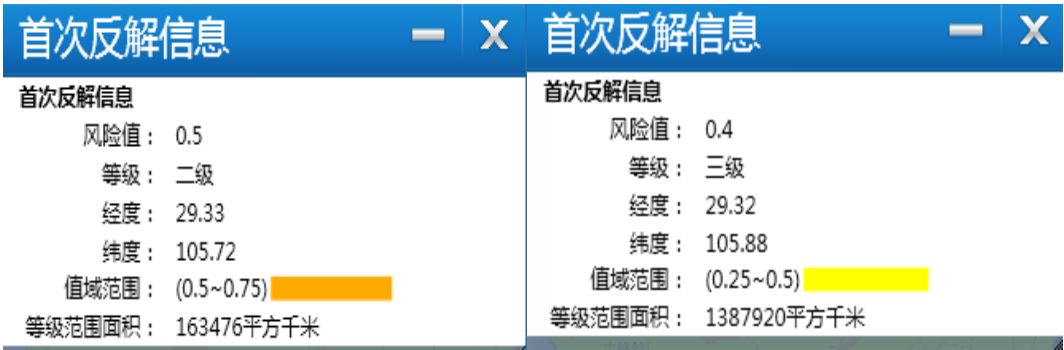


图 5.9 研判结果属性查询



图 5.10 勾选承载体后查询结果

一次研判结果的默认 4 个风险等级，各占 25%的比例，二次研判是在一次研判的基础上，调整各级所占比例，再进行研判。

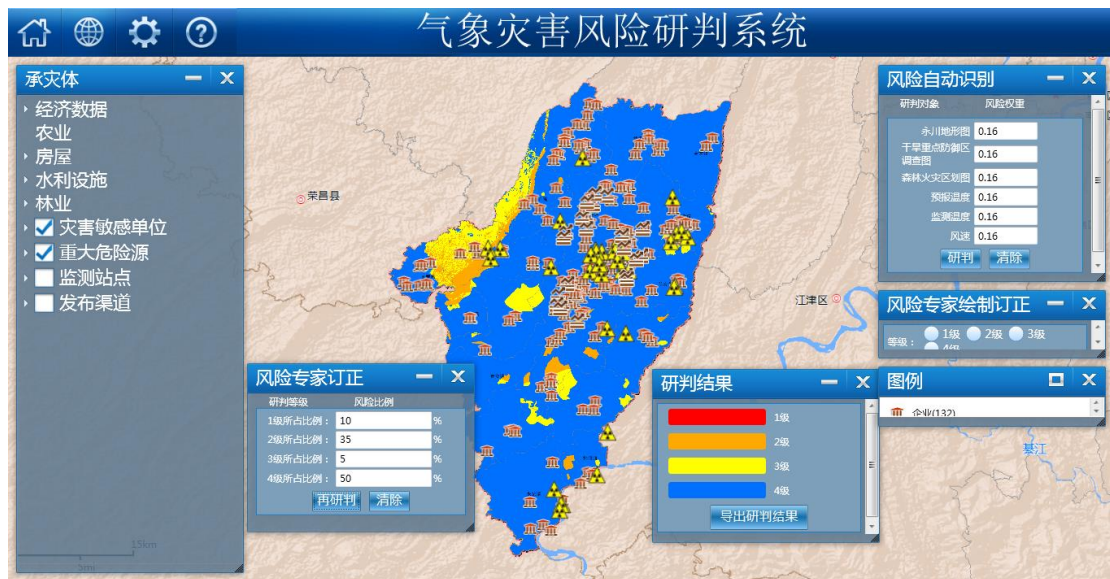


图 5.11 二次研判结果图

如果对二次研判结果还不满意，可进行绘制研判，选取风险等级和绘制形状后进行绘制，点击“绘制研判”再次生成研判结果

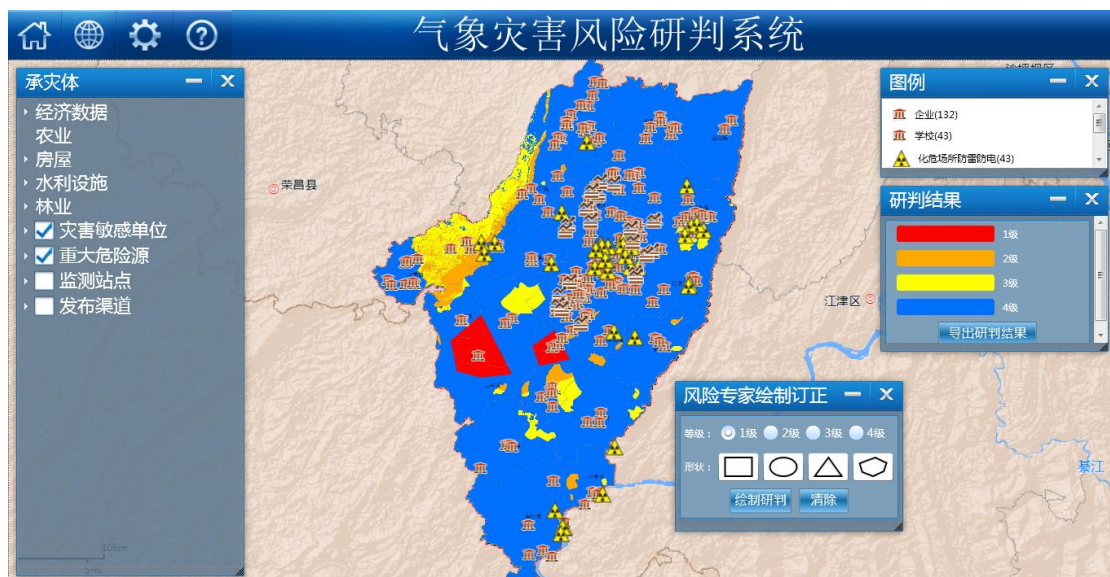


图 5.12 专家绘制研判结果图

本系统已经在重庆市主城区以及各个区县气象部门使用，验证了系统的通用性和稳定性。

5.2 效果评价

在 5.1 章节中，永川地形图是直接由永川 DEM 数据转换而来并存放在 Wr 文件中，经过与 DEM 数据比对，由 Wr 文件生成的永川地形图与 DEM 在 ArcMap 中生成的地形图完全匹配，说明了文件格式转换的正确性。春季低温区划图是通过灾害区划图制作模块由气象局获得相应的数据，把数据存放在 Wr 文件中，然后空间插值生成的春季低温区划图，区划范围与实际区划吻合度在 98%以上，说明了灾害区划模块的实用性、数据处理的严密性和正确性。雷达回波图是解析了新一代天气雷达数据，然后把雷达数据转化为 Wr 文件，并

出图，经过与原始雷达回波图比较，匹配率 100%，完全匹配，证明了雷达数据格式解析的正确性，同样的方法也验证了 Shapefile 文件和 IMG 文件解析的正确性。

5.1 章节中，把精细化预报温度和实时监测数据风速作为致灾因子，把干旱区重点防御调查数据和森林火灾区划数据作为孕灾环境进行风险研判。首先风险自动识别结果显示四个等级，大部分地区处于第三个级别，高危区没有出现，第二级别风险区出现在永川的西北部，此结果与当日气象部门监测数据相当吻合。专家根据经验对参与的研判因子进行权重重新分配，然后二次研判，权重的修改，使得研判因子的作用得到调正，可以由专家控制哪部分研判因子占的比例较大，研判出结果正确数据。专家还可以手动在研判结果上绘制相应等级的范围，来弥补漏判的区域。研判数据的属性查询显示没有企业出现在高风险区，25 个企业处在第二级别风险区，165 个企业和 20 个地质灾害点处在第三级别风险区，没有企业处在第四风险区。这个结果与当日实际情况完全吻合。用户可以选择向哪个级别的承灾体(企业)发送预警信息，使得这些承灾体提前做好防灾工作，把损失降到最小。通过本系统在重庆主城区和各个区县的推广应用和用户的反馈信息，已经使得本系统成为日趋成熟稳定的系统。

第六章 结论与展望

6.1 结论

本文利用气象监测实况数据、精细化预报数据、基础地理数据、灾情调查数据、社会经济数据等，采用自定义的文件格式 Wv, Wr 数据文件格式，利用 CSharp 语言设计开发了基于自定义文件气象灾害风险研判系统。系统包括了气象灾害调查图制作、风险灾害区划图制作以及气象灾害风险研判的重要模块。在充足的数据支持和经验丰富的专家协助研判下，本系统可研判多种气象灾害，精细化程度高，效果好，专题图美观大方，业务流程清晰规范。

本文完成的工作总结如下：

1. 精细解析了 ESRI 的 ShapeFile 数据文件、ERDAS 的 IMG 数据文件、新一代天气雷达数据文件，并在此基础上定义了新的数据文件 Wr 数据文件、Wv 数据文件、Pyr 数据文件，并实现了对各种数据文件的读写操作和数据文件的相互转换工作。
2. 本系统实现了 GIS 中经典的空间分析算法，并把这些算法成功移植到 Wr 文件和 Wv 文件格式中，自定义的 Pyr 文件实现了栅格数据的快速显示操作，Pyr 文件相当与栅格中的金字塔文件，本系统的数据源均是 Wr 文件和 Wv 文件，Wr 文件的快速显示为系统的用户良好体验提供了重要支持。
3. 总结了当前气象灾害风险研判的研究现状，基于气象灾害的成灾理论和研判方法，建立了气象灾害研判模型，并在此基础上设计和实现了气象灾害风险评估系统。
4. 本系统采用引用类库的方法调用实体对象和方法，用 CSharp 语言编写了 Model.dll 和 Common.dll 两个类库。Model.dll 类库中完成了自定义文件结构的设计和系统中使用的基本实体的定义，Common.dll 类库中实现了对自定义文件的基本操作和经典的 GIS 空间分析算法以及数据格式间的相互转换。其他数据格式到自定义文件格式的转换利用开源的代码，统一转换到自定义文件格式，这样间接的拓宽了系统的数据源。而且系统是直接引用 Model 和 Common 类库，减少了系统的代码量，使得修改维护起来方便快捷。
5. 以构件开发的方式设计和开发了基于 GIS 的气象灾害风险评估系统，开发周期短、效率高、可扩展性和移植性高，能在 Windows XP/Windows7 操作系统下稳定运行，后期维护更新方便快捷。
6. 设计开发了气象灾害风险调查图制作模块和风险灾害区划图制作模块，使得研究区域可以随意选择，提高了系统的通用性。
7. 系统在风险自动识别研判的基础上设计开发了专家二次研判和专家二次绘制研判功

能。根据专家经验对自动识别研判不满意之处可以重新设置各个研判等级的范围重新研判，还可以手动制定研判等级绘制区域，使得系统的研判误差更小，研判准确性更高。

8. 系统不经可以把各种调查图和灾害风险区划图以及研判结果保存为数据文件，还可以导出为专题图，还可以导出指定的风险等级内的承载体信息，并对这些信息做分类统计，根据需要通报给相关部门。
9. 本系统已经在重庆主城区及各个区县的气象部门运行适用，经过用户的反馈与维护更新，已经成为一个稳定、功能逐渐完善、流程清晰规范的业务化运行系统。能研判大部分气象灾害，有很好的实用性。

6.2 展望

气象灾害风险研判是一个复杂的过程，由于能力和时间问题，系统还有很多地方需要完善，将发现的不足之处总结如下：

1. 每一种气象灾害都有自己的运作机理，而且有些气象灾害也有很成熟的评估模型，所以系统可以试着实现多种模型，提供分析对比，取长补短，为用户提供不同模型的研判结果，使研判精度更高更准确。
2. 系统数据量大，查询起来费时较长，因此可以考虑使用新的空间索引方法，优化数据查询算法，实现对海量数据的快速查询和空间分析，提高系统查询效率。

参考文献

- [1] 刘昌杰. 基于 GIS 的气象灾害风险精细化评估系统研究与实现[D]. 南京信息工程大学, 2012.
- [2] 姜彤, 许朋柱. 自然灾害研究中的社会易损性评价[J]. 中国科学院院刊, 1996, (3):34~39
- [3] 张继权, 李宁. 主要气象灾害风险评价与管理的数量化方法及其应用[M]. 北京: 北京师范大学出版社, 2007.
- [4] 张俊香, 黄崇福. 自然灾害区划与风险区划研究进展[J]. 应用基础与工程科学学报, 2004(12):55-61.
- [5] 陈怀亮, 邓伟, 张雪芬, 等. 河南小麦生产农业气象灾害风险分析及区划[J]. 自然灾害学报, 2006, 15(1):135-143.
- [6] 黄崇福, 刘新立, 周国贤, 等. 以历史灾情资料为依据的农业自然灾害风险评估方法[J]. 自然灾害学报, 1998, 7(2):1-9.
- [7] 黄崇福, 张俊香, 陈志芬, 等. 自然灾害风险区划图的一个潜在发展方向[J]. 自然灾害学报, 2004, 13(2):9-15.
- [8] 张克中, 顾丽华, 万奎, 等. 德清县气象灾害风险区划技术方法研究及其应用[C]. 第26届中国气象学会年会公共气象服务与灾害管理分会场论文集, 2009:146-148.
- [9] BLAICKIE P, CANNON T, DA VIS I, et al. At risk:natural hazards people' svulnerability, and disasters[J]. London:Routledge, 1994:13-21.
- [10] PETAK W J, ATKISSON A A. Natural hazard risk assessment and publicpolicy[M]. New York;Springer-Verlag, 1992.
- [11] BURTONIR, KATESW, WHTE, GF. The environments hazard[M];(2ndseds.). New york:The Guildford Press, 1994.
- [12] 黄雁飞. 我国重大气象灾害应急管理体系研究: [硕士学位论文]. 上海: 上海交通大学, 2007.
- [13] 罗培. 区域气象灾害风险评估: [硕士学位论文]. 重庆: 西南师范大学, 2005.
- [14] 任清泉. 上海市水文气象灾害风险特征分析及管理范式研究:[硕士学位论文]. 上海:上海师范大学, 2007.
- [15] 王晓静. 广州主要气象灾害及其影响研究:[硕士学位论文]. 广州:广州大学, 2007.
- [16] 陈秀峰. 河北省气象灾害风险区划的气候评估技术研究:[硕士学位论文]. 兰州:兰州大学, 2010.
- [17] 刑开成. 气象灾害风险扩散机理及评估应用研究:[硕士学位论文]. 兰州: 兰州大学, 2011.
- [18] 杨丰政. 基于 GIS 的徐水县气象灾害风险评估研究:[硕士学位论文]. 南京: 南京信息工程大学, 2012.
- [19] 刘彤, 闫天池. 我国的主要气象灾害及其经济损失[J]. 自然灾害学报. 2011(2)
- [20] 刘维. 基于 RS 和 GIS 的江苏省重大农业气象灾害监测、预警系统的研究:南京:南京信息工程大学, 2012.
- [21] 黄崇福. 自然灾害风险评价理论与实践[M]. 背景:科学出版社, 2005.
- [22] 杨秋珍, 徐明, 李军. 对气象致灾因子危险诊断方法的探讨[J]. 气象学报, 2010, 68(2):277-284.
- [23] 张继权, 李宁. 主要气象灾害风险评价与管理的数量化方法及其应用[M]. 北京: 北京师范大学出版社, 2007:20-21.
- [24] 史培军. 灾害研究的理论与实践[J]. 南京大学学报, 1999, (11):37-42.

- [25] 白美兰等. 内蒙古典型农牧交错区孕灾环境特征及气象灾害风险识别[J]. 干旱地区农业研究, 2006, 24(4): 149-154.
- [26] 朱政, 刘仁义, 刘南. Img 图像数据格式分析及其超大数据量快速读取方法[J]. 计算机应用与研究, 2003, 3695 (8): 60-61.
- [27] 陈海泉, 廖仕湘. 新一代气象雷达产品数据格式分析[C]. 中国气象学会 2007 年年会论文集. 北京, 2007: 390-396
- [28] 王国华, 繆启龙, 宋健, 等. 杭州市气象灾害风险区划[M]. 北京: 气象出版社, 2012.
- [29] 陈家金, 王加义, 林晶, 等. 基于信息扩散理论的东南沿海三省农业干旱风险评估[J]. 干旱地区农业研究, 2010, 28(6): 248-253.
- [30] 丁燕, 史培军. 台风灾害的模糊风险评估模型[J]. 自然灾害学报, 2002, 11(1): 34-43.
- [31] 谭宗琨. 广西农业气象灾害风险评价及灾害风险区划[J]. 广西气象, 1997, 18(1): 44-50.
- [32] 易高流. 雷电监测资料在雷击损害风险评估中的应用[J]. 海洋预报, 1998, 15(3): 39-44.
- [33] 盛绍学, 马力文, 张晓煜, 等. 春小麦干热风灾害监测指标与损失评估模型方法探讨-以宁夏引黄灌区为例[J]. 应用气象学报, 2004, 15(2): 217-224.
- [34] 魏瑞江, 姚树然, 王云秀. 河北省主要农作物农业气象灾害灾损评估方法[J]. 中国农业气象. 2000, 21(1): 27-30.
- [35] 姜爱军, 郑敏, 王冰梅. 江苏省重要气象灾害综合评估方法的研究[J]. 气象科学. 1998, 18(2): 196-201.
- [36] 乐肯堂. 我国风暴潮灾害风险评估方法的基本问题[J]. 海洋预, 1998, 15(3): 39-44.
- [37] 冯定原, 吕新. 伊犁地区畜牧气象灾害灰色关联分析综合评估[J]. 南京气象学院学报. 1997, 20(1): 126-130.
- [38] 陈怀亮, 邓伟, 张雪芬, 等. 河南小麦生产农业气象灾害风险分析及区划[J]. 自然灾害学报, 2006, 15(1): 135-143.
- [39] 刘静, 马力文, 张晓煜, 等. 春小麦干热风灾害监测指标与损失评估模型方法探讨-以宁夏引黄灌区为例[J]. 应用气象学报, 2004, 15(2): 217-224.
- [40] 易高流. 雷电监测资料在雷击损害风险评估中的应用[J]. 江西气象科技, 2004, 27 (4): 45-47.
- [41] 张斌, 陈海燕, 顾骏强. 基于 GIS 的台风灾害评估系统设计开发[J]. 灾害学, 2008, 23(1): 47-50.
- [42] 樊敏剑, 王挣. 浅谈利用 3 S 建立洪水预警和灾害评估系统[J]. 内蒙古水利, 2006, 3: 22-23.
- [43] 吴浩云, 江南, 俞高宇. 基于组件式 GIS 的太湖流域洪涝灾害评估系统[J], 2001, 13(2): 127-134.
- [44] 刘俊, 徐向阳. 江苏省洪涝灾害评估系统设计[J]. 自然灾害学报. 1999, 8(4): 80-84.
- [45] 陈建. 贵州省旱情监测与灾害评估系统的分析与设计[D]. 云南: 云南大学, 2011.

致谢

时值本文完成之际，又是一年六月。三年的青海师范大学生活，平淡中感受着青春的忙碌。首先感谢我的导师曹广超教授，感谢曹老师对我硕士论文选题和写作的悉心指导，在本论文的写作过程中，我的导师曹广超老师倾注了大量的心血，从选题到开题报告，从写作提纲，到一遍又一遍地指出文中的具体问题，严格把关，循循善诱，在此我表示衷心感谢。同时，读硕期间，曹老师作为学者的严谨、长者的宽厚将让我终生记取。

感谢高小红教授、殷青军教授、沙占江教授等授过课的所有老师，是您们的知识浇灌着我成长；感谢研究生阶段的周薇薇、张艳娇、郭丽霞、蒋瑛、李金山、陶炳德、辛肖杰等同学的无私帮助和支持，很怀念和你们一起度过的三年硕士生活，在和你们的交流中获得的批评、建议和鼓励是我最大的财富，认识你们是我今生的幸福；感谢三年的舍友，一起撰写论文，互相鼓励与督促，使得毕业前那段颇有压力的时光增添了无限快乐，成为我非常值得怀念的记忆；还要感谢关心我的好友，是他们使我真切地体会到了友谊的真挚与绵长。

感谢陈露在我繁忙的工作学习中对我的理解和支持以及生活中的照顾，也祝愿陈露在以后的工作生活中一帆风顺、开开心心。

感谢我的爸爸妈妈爷爷奶奶，焉得谖草，言树之背，养育之恩，无以回报，你们永远健康快乐是我最大的心愿。

六月，芬芳馥郁。正如我周围一群风华正茂的学友，指点江山，直抒胸臆，青梅煮酒，纵论英雄。感谢同窗三载，学友对我生活和学习上的关怀和帮助。祝愿在以后的人生道路中，他们每个人都找到属于各自的幸福。

王孝萌

2013年6月6日

个人简历

个人信息			
姓 名	王孝萌	性 别	男
民 族	汉	出生年月	1988 年 01 月
籍 贯	河南开封	学 历	硕 士
毕业院校	青海师范大学	毕业时间	2013 年 6 月
学科专业	地图学与地理信息系统	研究方向	遥感与地理信息系统应用
联系地址	青海省西宁市五四西路 38 号青海师范大学生命与地理科学学院 (810008)		
政治面貌	中共党员	E-mail	695022058@qq.com

教育背景	
2010.09—2013.06	青海师范大学 生命与地理科学学院 地图学与地理信息系统 遥感技术应用 硕士研究生
2006.09—2010.06	郑州大学 地理信息系统 本科

参与科研课题和论文写作情况
2011 年 12 月 西北地区生态环境培训实践 2011 年 12 月 青海祁连山三河源区生态环境变化与质量评价 2012 年 5 月 黑河、布哈河、大通河源区环境动态变化与定量评价研究 论文 《Img 文件格式解析及读取》 科技风