

基于 Web 的文件系统信息展示方法

贾令涛¹, 李 丽²

(1. 中国飞机强度研究所, 陕西 西安 710065;
2. 西藏民族大学 信息工程学院, 陕西 咸阳 712082)

摘 要:对于存放于数据服务器中共享的数据文件信息,随着数据量的增加将逐渐难以共享、查阅和使用。为了便于共享这些数据资料,文中提出一种基于 Web 的文件系统信息展示方法,实现在 Web 页面上查找并展示数据文件。在分析数据文件存储方式的基础上提出九条定义,结合 SQL Server 设计灵活的文件信息存储关系表。设计递归算法从文件系统中读取指定目录下的目录和文件信息,利用 ASP.NET 的 TreeView 控件和 Table 控件的动态编程控制方式,设计递归算法动态生成 TreeView 和 Table 的数据显示表格,实现文件系统中目录及其文件信息的直观显示,并通过为所有文件提供名称检索及超链接下载的方式实现试验数据信息的共享。

关键词:文件系统;目录结构;ASP.NET;TreeView 控件;Table 控件

中图分类号:TP311.1

文献标识码:A

文章编号:1673-629X(2016)04-0066-04

doi:10.3969/j.issn.1673-629X.2016.04.014

A Method of File System Information Display Based on Web

JIA Ling-tao¹, LI Li²

(1. Aircraft Strength Research Institute of China, Xi'an 710065, China;
2. School of Information Engineering, Xizang Minzu University, Xianyang 712082, China)

Abstract: The huge amounts of data stored in servers are not convenient for sharing, searching and using. In order to share the test data widely, the file system, SQL Server and Web technique are integrated into the method which shows the data files on the Web. First, the data files and their relationship is analyzed, and nine definitions are formed. In succession, the flexible file store tables of SQL Server is constituted, and then a recursive arithmetic which obtains the information of specified directory in the disks is designed. Finally, using a recursive arithmetic by dynamically programming the TreeView and Table Control of ASP.NET, the directories and files stored in the data table could be showed on the Web, and the detail information of files could be obtained by opening the hyperlinks.

Key words: file system; directory structure; ASP.NET; TreeView Control; Table Control

0 引言

日常工作中经常需要将一些知识、经验和数据等以文件形式存放于数据服务器中,以实现资源的集中存储和共享。但随着数据文件的持续汇总,数据文件不仅需要更大的存储空间,同时目录结构也纷繁复杂,在增加管理难度的同时,导致从其中查找目标文件也越来越困难。而通过对数据文件存储方式的分析发现:数据文件以不同的目录名称保存在多级目录中,同时每一级目录中存在各种不同格式的数据文件,因此目录结构、文件格式和文件数量不具有任何规律性。

为了便于共享、查阅、使用数据文件,构建基于网络的管理系统以获取和管理资料是一种便捷的方法^[1-3];

通过递归算法获取文件夹的目录结构、属性和文件数据等信息以存储于数据库^[4-5],能够有效保留原始数据的分类和数据信息。结合上述两种思想,文中通过对文件系统^[6-7]、数据库技术^[8]以及 ASP.NET 技术^[9-10]的研究,提出一种基于 Web 的文件系统信息展示方法。

(1)在分析数据文件的目录与文件存储方式的基础上提出九条定义;

(2)在此基础上,设计一种灵活的文件信息组织与存储关系,使用存储过程(Stored Procedure)^[11-12]实现数据库表及各表之间关系的动态建立,并将提取的目录结构和文件信息实现分层存储;

收稿日期:2014-12-09

修回日期:2015-04-13

网络出版时间:2016-03-22

基金项目:中航工业技术创新基金(2013A62302R)

作者简介:贾令涛(1983-),男,硕士,工程师,研究方向为计算机应用、数据管理技术。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20160322.1518.022.html>

(3)设计递归提取算法将数据文件的目录、文件名称及其关系信息存储于关系数据库表中;

(4)从数据库表中读取存入的数据文件信息,并设计递归算法根据数据文件及其目录的关系动态生成 TreeView^[13] 和 Table,在 Web 页面中查询、使用数据文件^[14]。

1 数据文件组织与存储

为了实现数据文件及其相关关联信息的准确共享与展示,必须预先对这些信息进行组织并使存储方案的规划合理。结合文献[4-5]中的描述,在对数据文件系统深入分析的基础上,为了能够准确、完整地表述数据文件系统的信息,文中方法确定的数据文件组织与存储方案包括以下信息:

(1)目录结构和文件关系。描述数据文件与各级目录之间的从属关系、数据文件相关的各级目录之间的从属关系,以及各个目录的相关属性信息。

(2)文件属性信息。描述数据文件的相关属性信息,包括文件名称、类型、大小、创建日期等信息。

(3)文件路径。描述数据文件相对于根目录的相对存储路径信息,以确定各个数据文件路径。

(4)目录路径。描述各级目录相对于根目录的存储路径信息,以确定各个目录路径。

在上述基本信息的基础上:为了方便、准确地描述,针对数据文件的属性、目录类型及其属性等信息提出八条定义;为了准确记录八条定义中描述的数据信息及关联关系,以及相关的各类信息,设计相应的数据库表结构以满足数据文件在数据库中的表现形式需求;为了将数据文件相关的文件系统信息准确存储至数据库表中,设计了相应的递归算法在获取文件系统中相关目录及文件信息的同时存入数据库表中。

1.1 定义

- 定义 1:父目录,指当前目录的上一级目录;
- 定义 2:子目录,指当前目录的所有下一级目录;
- 定义 3:子文件,指当前目录下的所有数据文件;
- 定义 4:子孙目录,指当前目录的各级子目录;
- 定义 5:叶子目录,指子目录数为零的目录;
- 定义 6:目录层级,指当前目录所处层级,如根目录的目录层级为 0 级,根目录的子目录的目录层级为 1 级,之后逐级递增;
- 定义 7:目录距离,指当前目录与指定目录的目录层级之差;
- 定义 8:目录行数,指当前目录的子孙目录中叶子目录的数量,与当前目录是否存在子文件(存在为 1,不存在为 0)之和;
- 定义 9:目录深度,指当前目录与子孙目录中所有

叶子目录的目录距离最大值。

1.2 数据库表结构及数据库表创建

1.2.1 数据库表结构

由于文中所涉及的数据库表结构是连接文件系统中数据文件及其结构信息与 Web 页面之间信息展示的桥梁,因此对数据库表结构设计的两个最基本要求为:

(1)能够根据上述九条定义的描述,清晰、准确地记录数据文件相关的目录结构以及目录之间、文件与目录之间的关系;

(2)数据库表数据项中所记录的信息足够指导在 Web 页面上动态生成 Table 表格。

因此综合考虑上述两条要求,在数据库表设计时将同一目录下的子目录和子文件保存在同一张数据库表中,而子目录与子文件的具体属性信息由相应的若干数据库字段描述,同时通过相应的数据库字段描述子目录数、子文件数、目录深度和目录行数等信息。各个数据库字段信息简要描述如表 1 所示。

表 1 FileDirInfor 表结构

字段名	字段类型	字段描述
ID	Int	记录标识(记录唯一标识)
Name	Varchar	目录(文件)名
DorF	TinyInt	记录类型(目录或文件)
Path	Varchar	目录或文件的相对路径
KeyWords	Varchar	描述文件的关键词
ParentID	Int	父目录的记录标识
SFCCount	Int	子文件数目
SDCount	Int	子目录数目
DepthLevel	Int	目录深度
RowCount	Int	目录行数

表 1 给出了数据库表结构的各个字段的简要描述信息。这些字段不仅记录了各个数据文件与目录的基本属性信息,同时实现数据文件、目录及其与父目录、子目录之间关联关系的清晰记录。下面针对数据库表结构中各个字段的含义以及各个字段与设计要求之间的关联关系进行具体描述:

(1)不同目录层级之间单独使用数据库表记录,因此每一层级的表名为 Table_{*i*},其中 *i* 表示该表所记录的目录层级,如根目录表名为 Table₀。

(2)ID 是当前记录在其父目录下的唯一标识。

(3)Name 描述当前记录的名称。

(4)DorF 描述当前记录是目录信息或文件信息。0 为目录,1 为文件。

(5)Path 描述当前目录或文件相对于根目录的相对路径信息。

(6)KeyWords 仅当 DorF 为 1 时有效,是对数据文件内容信息的关键字描述。

(7)ParentID 表明该条记录与上一层级数据库表中的哪条记录的目录间存在父子关系。

(8)SFCount 描述当前目录下子文件的数目(若该记录 DorF 为 1,即为数据文件时,该值为 0),并为父目录的 RowCount 字段提供数据支持。

(9)SDCount 描述当前目录下子目录的数目(若该记录 DorF 为 1,即为数据文件时,该值为 0),为 0 则表示该目录为叶子目录。

(10)DepthLevel 描述当前目录的目录深度(若该记录 DorF 为 1,即为数据文件时,该值为 0),也即当前目录至根目录的目录距离,同时该字段可以指明 Table 控件的列数。

(11)RowCount 描述当前目录下叶子目录的数量和子文件的数量信息(若该记录 DorF 为 1,即为数据文件时,该值为 0)。若当前目录存在子文件则在其叶子目录数量基础上加 1,否则不操作,同时该字段与 Table 控件的行数对应。

根据上述描述易知,(1)~(11)条可以满足数据库表结构设计两个基本条件的要求,其中(10)和(11)所记录的数据信息用于后续动态生成 Table 控件。

1.2.2 文件系统信息递归提取算法

数据文件信息以目录和数据文件结合的方式组织和存储,包括若干目录及其目录结构、不同格式的数据文件。对此结构分析可知:

(1)各个目录的子目录数不具有任何规律,目录深度也不确定;

(2)当某个目录不存在子目录(即该目录为叶子目录)时其目录深度为 0;

(3)某个目录的目录深度为其子目录中目录深度最大值加 1。

在上述三条基本信息的基础上,文中方法通过设计递归算法在提取目录和数据文件基本属性信息的同时,获取目录与数据文件的层级结构信息,以支持后续基于 Web 页面的 Table 控件展示。给定一个目录,具体递归算法流程如下:

步骤 1:获取当前目录的目录层级数,如果大于已记录的目录深度,则更新目录深度为当前目录层级数,同时为当前目录创建新的数据库表,记录其基本属性信息。

步骤 2:判断当前目录的目录层级数,如果目录层级数为 0,则为根目录,其父目录标识为 0;如果目录层级数非 0,则为当前目录记录其父目录标识。

步骤 3:提取当前目录的子文件数,用于更新父目

录的子文件数。对于每一个符合要求的数据文件,将其基本属性信息存储于当前目录层级对应的数据库表中。

步骤 4:提取当前目录的子目录数,如果子目录数为 0,则表示该目录为叶子目录,进入步骤 5;否则,对于当前目录的每一个子目录,执行步骤 6。

步骤 5:对于叶子目录,设置其子目录数为 0,行数设置为 1,并将其父目录的叶子目录数目、行数分别加 1。

步骤 6:对于非叶子目录,根据子文件数、叶子目录数和子目录返回的目录行数更新当前目录的目录行数,即:如果子文件数为 0,则其目录行数等于返回的目录行数与叶子目录数之和;如果子文件数非零,则其目录行数设置为子目录返回的目录行数与叶子目录数之和加 1。

步骤 7:更新每一层级数据库表中每条记录的目录深度,即根据返回的最大目录级数更新每一条目录记录的目录深度。

2 数据文件信息展示

2.1 动态创建 TreeView 目录信息

数据文件以多级目录的形式存在,上述数据库表结构中记录的信息以若干根目录结构为基础,扩展为不同的数据文件及其目录信息,因此以这些根目录为基础,可形成若干目录树。不同目录树的子目录可能全部为子目录,也可能包含子文件,因此对于尚未包括子文件的目录树,将其设置为数据文件分类的参考,根据用户需要设置目录树的层级限定,通过递归算法创建目录结构,根据数据库中的信息自动生成 TreeView 控件。

算法流程如下:

步骤 1:从数据库中获取当前目录层级的目录信息。

步骤 2:对于每一个目录,分别从数据库中获取其子文件数。如果子文件数非零,则当前目录可作为 TreeView 的叶子节点,直接返回;否则,进入步骤 3。

步骤 3:子文件数为零,则获取其所有子目录,执行步骤 1 操作。

通过以上算法的递归执行,将数据文件的目录结构信息映射为 ASP.NET 的 TreeView 结构。该结构可以作为目录信息导引用户使用,当用户需要查看某目录下的详细子目录和数据文件时,可点击相应 TreeNode 的叶子节点,以查看其详细信息。

2.2 动态创建 Table 目录与文件详细信息

上一节根据用户设置不同的目录树层级可以在 Web 页面上依托 TreeView 控件形成相应的目录导航,

实现一定层级范围内的目录结构展示。为了进一步查看该目录导航下的子目录及数据文件信息,用户需进一步点击 TreeView 中相应的叶子节点。

对于 TreeView 中不同的叶子节点,以其为根目录能够重新构建各自子目录与子文件的树状结构。对于不同的叶子节点,由于其对应的目录标识所对应的目录深度和子目录数等信息不同,为了便于统一在 Web 页面展示,文中方法同样采用递归算法进行处理,根据该叶子节点包含的目录标识信息直接动态创建该目录对应的目录与文件信息的详细表格。

算法流程如下:

步骤 1:从数据库表中获取指定叶子节点对应目录的数据信息,根据标记(初始为否)判断是否需要新建 TableRow。如果为非叶子目录,进入步骤 2;否则,进入步骤 5。

步骤 2:为当前目录新建一个 TableCell,该 TableCell 的 RowSpan 为该目录的 CountOfRows、ColumnSpan 为 1,将其加入父目录所在的 TableRow。如果当前目录下子文件数不为 0,则进行步骤 3;否则,进入步骤 4。

步骤 3:为子文件新建一个 TableCell,该 TableCell 的 RowSpan 为 1、ColumnSpan 为当前目录的 DepthLevel,将其加入当前目录所在的 TableRow。从数据库中提取当前目录的所有子文件,为每一个子文件新建 HyperLink 与之对应,并将该 HyperLink 加入 TableCell。进入步骤 4。

步骤 4:对于当前目录的第一个子目录,新建 TableRow 的标记为 False,对于其他子目录,分别新建 TableRow 并加入 Table 后,进入步骤 1 进行递归处理。

步骤 5:为当前叶子目录新建一个 TableCell,该 TableCell 的 RowSpan 为 1、ColumnSpan 为 1,将其加入父目录所在的 TableRow。同时再新建一个 TableCell,该 TableCell 的 RowSpan 为 1、ColumnSpan 为当前目录的 DepthLevel,将该叶子目录中的所有子文件从数据库中提取出来,为每一个子文件新建 HyperLink 与之对应,并将该 HyperLink 加入该 TableCell。

3 应用实例

如图 1 所示,通过文中方法将文件系统中指定目录下的数据文件及其目录关系信息显示在 Web 页面中。输入为“文件资料”根目录下的目录及相应的数据文件,在“文件资料”根目录下存在不同的子目录以及相应的数据文件。通过文中方法递归获取目录结构与数据文件信息后存储至数据服务器,之后通过 Web 服务器实现这些数据文件在 Web 页面的共享与显示,同时支持用户的关键词检索及下载使用。

图 1 中,左侧的 TreeView 控件中显示的是限定为

三级的目录树,右侧表格中显示的内容为左侧目录对应的子目录及其下的数据文件名称及其超链接,并可通过超链接查看具体的数据文件。

■ 文件资料	
□ 娱乐	
■ 学习	
■ ASP	学习——ASP——电子书
□ 代码示例	ASP+ACCESS 2000网站建设实训.pdf
□ 文章	asp_net_word.doc
□ 电子书	ASP数据库系统开发完全手册.pdf
□ 网页	中文版EXCEL2007高级VBA编程宝典.rar
■ JAVA	
■ Matlab	ASP.NET+SQL数据库案例精萃.pdf
■ Oracle	ASP.NET 2.0入门经典(第四版)源码.rar
□ SQL SERVER	ASP.NET 3.5开发范例精讲精析.pdf
□ 其他资料	ASP.NET从入门到精通.pdf
□ 数据库文档	ASP.NET完全入门.rar
□ 工作	More...
	new_page.doc
	path.txt
	VB deal with Word.doc
	SpecifiedFileHandler.rar
	ASP.NET完全入门.rar
	More...

图 1 应用示例

4 结束语

通过对文件系统、数据库技术以及 ASP.NET 技术的研究,设计并实现了一种基于 Web 的文件系统信息展示方法,将大量数据文件通过 Web 形式展示,便于数据的共享、查找和使用。在分析文件系统中目录与文件存储方式的基础上,设计了一种灵活的文件信息存储关系,并通过递归提取算法将文件系统中目录和文件信息存储于关系数据库表中,在读取数据库信息后动态递归生成 TreeView 和 Table,在 Web 页面中将指定目录的详细文件信息显示出来并提供各个数据文件的超链接。该方法可以十分便捷地将目前存储在硬盘文件系统中的数据信息显示于 Web 页面,便于用户对数据信息的查阅、共享、使用,实现了数据信息的资源共享。

参考文献:

- [1] Jones W, Bruce H, Dumais S. Keeping found things found: the study and practice of personal information management[M]. Boston: Morgan Kaufmann Publishers, 2008.
- [2] 陈定权, 刘颖. 参考文献管理软件评析与展望——以 EndNote、NoteExpress 为例[J]. 现代图书情报技术, 2009(7): 80-84.
- [3] 张 媚, 黄 穗, 邓彩细. 面向科研团队的 Web 文献协作管理系统构建[J]. 微计算机应用, 2010, 31(7): 58-62.
- [4] 李昌贵, 吕志平. 数据库中文件夹的整体存储与随机访问[J]. 计算机工程, 2011, 37(5): 41-43.
- [5] 李 良, 柴 毅, 王道斌. 基于 .NET 的 Oracle BLOB 数据高效存取方法[J]. 计算机工程, 2008, 34(20): 64-65.
- [6] 居锦武, 王兰英. NTFS 文件系统剖析[J]. 计算机工程与设计, 2007, 28(22): 5437-5439.
- [7] Custer H. Inside the windows NT file system[M]. [s. l.]: Microsoft Press, 1994.
- [8] Stephens R. 数据库设计解决方案入门经典[M]. 北京: 清

(下转第 73 页)

表示,结果分别如图5(d)、(e)所示。

3 实验结果与分析

为了验证文中车标区域定位方法是否有效,这里对车脸定位后的1258幅车脸图像进行了实验。实验中采用的统一阈值为 $T_h = T_v = 420$, $T_x = 330$,统计结果见表1。

表1 检测结果

	数目	概率/%
实际检测到的车标区域	1 076	85.53
未检测到的车标区域	182	14.47
检测到的正确车标区域	943	87.64
检测到的错误车标区域	133	12.36

根据表1可见,车标定位结果存在误判和漏判。引起误判的主要原因是车标斜向纹理表现不明显,并且车辆散热隔栏器具有跟车标相似的横竖纹理,如图6上图所示;而引起漏判的主要原因是车标的亮度比背景亮度低很多,又不满足上述规定的阈值,则车标区域定位失败,如图6下图所示。但如果降低阈值又容易产生误判,因此,采用自适应阈值的方法将是下一步的研究方向。



图6 误判和漏判结果

4 结束语

针对交通道路环境下车标快速定位的问题,文中先采用Adaboost算法对车辆对象进行车脸区域的定位,减少道路背景以及车辆对象非车脸区域的纹理影响,然后直接在车脸区域的DCT域提取车标区域的纹理特征,利用车标具有的斜向纹理特征来区别于车脸区域的其他背景部分,从而达到定位车标的目的,充分用了DCT域数据的特点。实验证明,该方法能够有效地实现车标区域的定位,为交通卡口抓拍的车牌无

法识别车辆的车标识别提供了一种新思路。

参考文献:

[1] Wang Y, Liu Z, Xiao F. A fast coarse-to-fine vehicle logo detection and recognition method [C]//Proc of IEEE international conference on robotics and biomimetics. [s. l.]: IEEE, 2007:691-696.

[2] Liu Yang, Li Shutao. A vehicle logo location approach based on edge detection and projection[C]//Proc of IEEE international conference on vehicular electronics and safety. [s. l.]: IEEE,2011:165-168.

[3] 李玲. 车标定位方法研究[D]. 大连:辽宁师范大学, 2009.

[4] 李红林,王运琼. 基于差分与对称性检测相结合的车标定位方法[J]. 曲靖师范学院学报,2008,27(6):68-71.

[5] 李贵俊,刘正熙,游志胜,等. 一种基于熵增强和自适应形态学滤波的快速车标定位方法[C]//第三届信号与信息处理全国联合学术会议论文集. 出版地不详;出版者不详, 2004.

[6] 肖飞,王运琼,刘丽梅,等. 基于局部对称性特征的快速车标定位[J]. 计算机科学,2010,37(2):298-300.

[7] 马蓓,张乐. 基于纹理特征的汽车车型识别[J]. 电子科技,2010,23(2):94-97.

[8] 李欣昊. 智能交通系统中车辆检测关键技术研究[D]. 长春:吉林大学,2011.

[9] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting[J]. Journal of Computer and System Sciences,1997,55(1):119-139.

[10] Schapire R E, Singer Y. Improved boosting algorithms using confidence-rated predictions[J]. Machine Learning,1999,37(3):297-336.

[11] 李全武,李玉惠,李勃,等. 车脸定位及识别方法研究[J]. 计算机科学与探索,2015,9(6):726-733.

[12] Chiptraser B, Rao K R. Discrete cosine transform filtering [J]. Signal Processing,1990,19(3):233-245.

[13] Ramsubramanian D, Venkatesh Y V. Encoding and recognition of faces based on the human visual model and DCT[J]. Pattern Recognition,2001,34(12):2447-2458.

[14] 黄祥林,沈兰荪. 基于DCT压缩域的图象字符定位[J]. 中国图象图形学报,2002,7(1):22-26.

(上接第69页)

华大学出版社,2010.

[9] 瞿杰. Programming ASP.NET 中文版[M]. 北京:电子工业出版社,2007.

[10] Kauffman J, Matsik B. Beginning ASP.NET databases using C#[M]. [s. l.]: Wrox Press Ltd., 2002.

[11] Mourad O, Athman B. Query processing and optimization on the web[J]. Distributed and Parallel Databases,2004,15(3):187-218.

[12] Tan B, Zeng L. A performance optimization based on stored procedure in RDBS project[C]//Proc of international conference on computer & communication technologies in agriculture engineering. [s. l.]: [s. n.], 2010:594-597.

[13] 杜娟,杨玮清. TreeView 目录构建及数据拖放的编程开发[J]. 数字技术与应用,2011(11):148-148.

[14] 周炎涛,陈贤谋. ASP.NET 中 TreeView 控件与数据库结合创建动态目录树[J]. 航空计算技术,2004,34(2):25-27.