JSPM's
Rajarshi Shahu College of Engineering
Department of Computer Engineering
[2012-2013]

PROJECT ASSIGNMENTS

TITLE OF PROJECT:

KWEST: A Semantically Tagged Virtual File System

GROUP NO. : 10

GROUP MEMBERS:

Sr. No.	Name of student	Exam Seat No.
1	Aseem Gogte	B80374220
2	Sahil Gupta	B80374222
3	Harshvardhan Pandit	B80374244
4	Rohit Sharma	B80374256

SIGNATURE OF GUIDE :

NAME OF GUIDE **:** V. V. Phansalkar

INDEX

1. The modules of the project

1.1. Module 1: File system operations (FUSE operations)

This module interacts with FUSE to perform file system operations. The operating system expects operations to return results in a specific format. It is the responsibility of this module to convert query results to this format before sending it back to FUSE.

1.2. Module 2: Generating Tags

This module is responsible for generating appropriate tags for files. Common file types like images, audio have a fixed set of attributes under which the files are tagged. These can be extracted by using various external libraries such as TagLib, EXIF. Default tags are generated for files if metadata cannot be extracted or does not exist. User and system defined rules can be used to associate related files and generate tag accordingly.

1.3. Module 3: Importing Semantics

Users already have certain organizational structures in the way they store data in file systems. This module imports semantics by converting the storage hierarchy to tag-based hierarchy. This means the directory structure present in the file system will be used to form tags and the files listed under the directory are tagged under that tag.

1.4. Module 4: Exporting Semantics

This module can export the storage hierarchy to some external location. The semantic organization of tags is converted to actual directories and the files are then copied to these directories. This is similar to copying contents from one file system to another.

1.5. Module 5: Database Consistency

It is vital for the proper functioning of the system that the database always remains consistent. Logging mechanisms ensure that operations on the database always reach an endpoint. This module is used to check, correct and maintain integrity of the database by checking for redundant entries. Also, if there are new files which have not been added to Kwest, this module can help the user add them.

2 The Domain Specific Analysis

2.1 Domain description: Semantic File Systems

Files are a popular form of data storage as they do not impose any formatting and structuring constraints, and the file/directory abstraction is easy to understand. Conversely, the fact that data is stored in files provides no semantic information about them. Only tools that manipulate files carry much of the knowledge about file semantics. Semantic File Systems are file systems used for information persistence which structure the data according to their semantics and intent, rather than the location as with current file systems. It allows the data to be addressed by their content and querying for the data.

2.2 Creating and Managing Virtual Filesystems

A virtual file system (VFS) or virtual filesystem switch is an abstraction layer on top of a more concrete file system. The purpose of a VFS is to allow client applications to access different types of concrete file systems in a uniform way. A VFS can, for example, be used to access local and network storage devices transparently without the client application noticing the difference. It can be used to bridge the differences in Windows, Mac OS and UNIX filesystems, so that applications can access files on local file systems of those types without having to know what type of file system they are accessing.

File system in User space is a loadable kernel module for Unix-like computer operating systems that lets non-privileged users creates their own file systems without editing kernel code. This is achieved by running file system code in user space while the FUSE module provides only a ‘bridge’ to the actual kernel interfaces. FUSE is particularly useful for writing virtual file systems. Unlike traditional file systems that essentially save data to and retrieve data from disk, virtual filesystems do not actually store data themselves. They act as a view or translation of an existing file system or storage device.

2.3 Meta-Data

Metadata (meta content) is defined as data providing information about one or more aspects of the data, such as

- Means of creation of the data
- Purpose of the data
- Time and date of creation
- Creator or author of data
- Location on a computer network where the data was created
- Standards used

Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource. Metadata is often called data about data or information about information.

Some categories of meta-data are listed below:

Metadata can be stored either internally, in the same file as the data, or externally, in a separate file. Metadata that is embedded with content is called embedded metadata.

A data repository typically stores the metadata detached from the data. Both ways have advantage and disadvantages:

- Internal storage allows transferring metadata together with the data it describes; thus, metadata is always at hand and can be manipulated easily. This method creates high redundancy and does not allow holding metadata together.
- External storage allows bundling metadata, for example in a database, for more efficient searching. There is no redundancy and metadata can be transferred simultaneously when using streaming.

2.4 Reusable components

1. Meta data Repository

   Kwest stores all meta-data associated with a file in Relational database which is available for access to other programs.

2. Metadata extraction Module

   Kwest makes use of various external libraries to extract metadata for a number of file types. Programmers may directly use these modules to generate meta data without having to know the underlying details.

3. Semantic Repository

   Kwest uses rules like semantic navigation to relate various files based on context. Semantic relations based on rules are stored in a Relational database which provides a set of related files.

2.5 Domain Specific Language

SQL or Structured Query Language is a special-purpose programming language designed for managing data in relational database management systems (RDBMS). Originally based upon relational algebra and tuple relational calculus, its scope includes data insert, query, update and delete, schema creation and modification, and data access control.

3 Comparative study of various options available to implement the project modules

3.1 Creating and Managing Virtual Filesystem

Options available:

1. The Filesystem in Userspace (FUSE) mechanism allows userland code to plug into the virtual file system mechanism in Linux, NetBSD, FreeBSD, OpenSolaris, and Mac OS X.

2. Userland Shell namespace extensions allow virtual filesystems to be implemented on Microsoft Windows.

3. KIO and GVFS/GIO provide virtual file system mechanisms in the KDE and GNOME desktop environments (respectively). They can be made to use FUSE techniques and therefore integrate smoothly into the system.

4. The VFS tools and library are a collection of Functions, Structures and Tools, with which you can create and work with VFS on Windows.

5. Using Dokan library, we can create file systems very easily without writing device driver. Dokan Library is similar to FUSE (Linux user mode file system) but works on Windows.

6. Callback File System is a component set for presentation of data as files and directories of a local virtual disk. Regardless of where the actual data are kept: in files, in database records, memory, or elsewhere - they will be treated as if they were parts of a single file system.

7. LUFS is a hybrid userspace file system framework supporting many exotic file systems (localfs, sshfs, ftpfs, gnutellafs, locasefs, gvfs, cardfs, cefs, etc.).

Selected: FUSE (File System in Userspace)

With FUSE it is possible to implement a fully functional filesystem in a userspace program. Features include:

- Simple library APISimple library API

- Simple installation (no need to patch or recompile the kernel)
- Secure implementation
- Userspace - kernel interface is very efficient
- Usable by non-privileged users
- Runs on Linux kernels 2.4.X and 2.6.X
- Has proven very stable over time

3.2 Storing and managing Metadata

Options available:

1. SQLite is a relational database management system contained in a small C programming library. In contrast to other database management systems, SQLite is not a separate process that is accessed from the client application, but an integral part of it.

2. Firebird is an open source SQL relational database management system that runs on Linux, Windows, and a variety of UNIX.

3. MySQL is the world's most used open source relational database management system (RDBMS) as of 2008 that run as a server providing multi-user access to a number of databases.

4. H2 is a relational database management system written in Java. It can be embedded in Java applications or run in the client-server mode. The disk footprint (size of the jar file) is about 1 MB.

5. Microsoft SQL Server Compact is a compact relational database produced by Microsoft for applications that run on mobile devices and desktops.

6. Berkeley DB (BDB) is a software library that provides a high-performance embedded database for key/value data. As of 2012, Berkeley DB is the most widely used database toolkit in the world.

7. NoSQL database management systems are useful when working with a huge quantity of data and the data's nature does not require a relational model for the data structure.

8. The Oracle Database (Oracle RDBMS) is an object-relational database management system (ORDBMS) produced and marketed by Oracle Corporation.

Selected: SQLite

Unlike client-server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program. SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross- platform file on a host machine. Features include:

- Zero Configuration
- Serverless
- Single Database File
- Stable Cross-platform database
- Manifest typing

3.3 Extracting Metadata

Options available:

1. The libexif C EXIF library is a library written in pure portable C. It reads and writes EXIF meta information from and to image files.

2. GNU Libextractor is a library used to extract meta data from files. GNU libextractor uses helper-libraries (plugins) to perform the actual extraction. As a result, GNU libextractor can be extended simply by installing additional plugins. Currently, libextractor supports the following formats: HTML, MAN, PS, DVI, OLE2 (DOC, XLS, PPT), OpenOffice (sxw), StarOffice (sdw),

FLAC, MP3 (ID3v1 and ID3v2), OGG, WAV, S3M (Scream Tracker 3), XM (eXtended Module), IT (Impulse Tracker), NSF(E) (NES music), SID (C64 music), EXIV2, JPEG, GIF, PNG, TIFF, DEB, RPM, TAR(.GZ), LZH, LHA, RAR, ZIP, CAB, 7-ZIP, AR, MTREE, PAX, CPIO, ISO9660, SHAR, RAW, XAR FLV, REAL, RIFF (AVI), MPEG, QT and ASF. Also, various additional MIME types are detected.

3. The Metadata Extraction Tool was developed by the National Library of New Zealand to programmatically extract preservation metadata from a range of file formats like PDF documents, image files, sound files Microsoft office documents, and many others.

4. TagLib is a library for reading and editing the meta-data of several popular audio formats. Currently it supports both ID3v1 and ID3v2 for MP3 files, Ogg Vorbis comments and ID3 tags and Vorbis comments in FLAC, MPC, Speex, WavPack TrueAudio, WAV, AIFF, MP4 and ASF files.

Selected: to be decided after testing

We will first test the various alternatices with our system to see which works best. The given options list the choices we have for extracting metadata from files. Each has advantages and drawbacks. Testing will be done on the following criteria:

- Integration with system components
- Performance - speed, resources, etc.
- Safety and integrity of data
- Extraction capabilities for particular file types
- Modularity
- Legal requirements

It may be possible that we may require using separate libraries for different file types. In such cases, the preferred library would be the one which performs best for that particular category.