VfsNext

VFS - Next

done - 1. apply http://issues.apache.org/bugzilla/show_bug.cgi?id=33795

ram filesystem

partially done. 2. apply http://issues.apache.org/jira/browse/VFS-43

services & svn filesystem (we have to setup an external space to provide the svn filesystem as the used library JavaSVN is LGPL)

we need a discussion about a new name instead of "services" (maybe).

The idea is to have calls like

```
Commit commitFile = FileObject.getService(Commit.class)
commitFile.process();


TumbnailImage thumb = FileObject.getService(TumbnailImage.class)
thumb.process();
Image bi = thumb.getImage();


ExifData exif = FileObject.getService(ExifData.class)
exif.process();
String camera = exif.getCameraModel();
```

You get the idea?

Finally I decided to use the wording "operation" (idea from: http://people.tryphon.org/~alban/io-operations/docs/api/index.html) instead of "service".

3. user api

a way to avoid to put username/password into the url.

Using a callback mechanism to get this data when needed.

4. progress, pause and cancellation api

callbacks to be informed about ongoing operations (copy, move, rename, delete, copy-progress, move-progress (if between filesystems)) Ability to pause or cancel the operation.

5. browse roots

e.g. smb or local partitions (windows - c:, d:, ...)

done - 6. caching review - allow to configure fileObjects internal state cache

see VfsCacheStrategy

This is to avoid the .close() calling to get fresh data.

Sure, this might slow down alot, but if one would like have it, it should be possible.

Also try to find a way to avoid refreshing the internal children chache, or at least avoid successive refresh as good as possible.

7. xpath

allow xpath style filesystem browsing

done - 8. review virtual filesystem

see what happend to the virtual filesystem

9. file permissions (non-native)

(with non native code only)

figure out a way to represent file permissions and a way to modify then

10. native parts (if possible through jdic)

- filesystem notifications - to get informed about external filesystem events
- file permissions to allow fine grained setup for local files
- partition information (type, space, label) - some available in JDIC but also keep in mind this: http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=4057701

11. write support for archives

this would require it to

1. copy the archive locally
2. apply the change
3. repack the archive
4. copy the archive back to its place

We should stay in state 2. until one sends a signal to finish with the archive.

---

new filesystems

- pop, imap (useable for emails)
- mime (for the email itself - pretty much like an archive)

Consider this url:

```
zip:mime:imap://user@mailserver/INBOX/folders/newMail.mime!/attachment1.zip!/zipentry.txt
```

Cool, isnt it?

- generic XML
- ant (only a variant of XML? Cant we just set a dtd or scheme using some sort of configurationOption ?)
- java structure (by class, by source?)

Todos

The following is a list of items that need to be completed. Contributions are welcome!

Release 1.1

- moveTo should handle a directory move if it moves from a different filesystem.

Open

- More documentation (status, file naming etc).

- Fix the TODO items
- Add more providers:
    - rsync
    - subversion
    - nfs
    - cvs
    - jdbc filesystem
    - xml filesystem
    - jndi
    - imap
    - local mirror
    - spidering http
    - ...
- JNDI integration.

- Formalise the provider API.
- WebDAV Provider:
    - Add plain http support, and auto-detect dav resources.
    - Add set last-modified.
    - HTTPS
- Zip/Jar Provider:
    - Extract an AbstractLayerFileSystem out of ZipFileSystem.
    - Track changes to the parent layer. Eg when the parent layer is deleted, mark all the files in the fs as 'does-not-exist'.
    - Add support for writing to zip/jar files.
- URL Provider:
    - Support attributes.
- HTTP Provider:
    - Support attributes.
    - HTTPS support.
- The local disk caching mechanism also needs more work. Needs to check last-modified time. Replicator needs to be more configurable.
- Add persistent replicator.
- Finish support for junctions: Make ancestors of a junction point visible, fire events when junction is added or removed, tests.
- Add support for federation (ie transparently crossing file system boundaries, such as drilling down into the contents of a Jar file).
- Add an equivalent of the fileScanner Jelly tag.
- Add an equivalent of Ant path, fileset, dirset, filelist, etc. Ideally, these can be abstracted into a single data type.
- Allow selectors, name mappers, and filters to be specified for the Ant tasks.
- Add capabilities to FileObject.
- Attributes and attribute schema.
- Handle file canonicalisation better (for cases like case-insensitive file systems, symbolic links, name mangling, etc).
- Add more selectors: XPath, Ant style, regular expression.
- Add adaptor (NodePointerFactory?) for use with JXpath.
- Add content-changed, attribute-changed, and move events to FileListener. Maybe split into structure and content listeners.
- Get/set the file permissions.
- Automatically checksum and/or verify remote files.
- Look at adding native code for fine-grained control over permissions, file monitoring, faster moves, etc. Must be optional - the thing should still build and run without the native code.

Library upgrades

The following describes changes in upgrades dependencies which need to be considered:

webdavlib 2.1

- resource.listWebdavResources() do no longer list directories?
  Figure out what to do

wishes

The best chance to get those added is by contribute them 😉

Eclipse enabled VFS JARs

In addition to the current plugin model, use the OSGi plugin model to allow configuration of e.g. additional filesystem providers.

---

non confimed

Attribute API. This is useful for things like storing attributes of XML element in the XML filesystem (I did it but it is not standard), JNDI VFS (if I do it), or more simply, to retrieve infos like access rights (rwx) or file ownership in (ext2) filesystems. Such an API would allow an WebDAV server to be implemented on-top of a commons VFS.

a VFS on top of apache's configuration package

Spring-friendly

A Spring-friendly configuration interface, that allows file system managers to be fully configured as Spring beans. Such an interface would allow also junctions to be specified.

Support For Writing ISO-9660 and Joliet images

It'd be nice to be able to construct a CD or DVD layout using file system operations and then "burn" the resulting layout to an image file.

Support For Hierarchical Storage Management

It'd be nice to be able to add hierarchical storage management capabilities to the VFS.

(add your wishes here)

x. filesystem xyz

blablabla