FileXfer
File Transfer Jobs

Raymond E. Marcil

<marcilr@gmail.com>

Revision 0.0.1 (June 30, 2016)

Abstract

The FileXfer application is a system for automated file transfer jobs for copying files. "There are 3 applications that make up the usage collection framework: `filexfer`, which does the actual file transfers; `filexfer-jobmonitor`, which is configured to monitor various aspects of jobs and create NMS alarms when necessary; and `filexfer-dataloader`, which bulk-loads file data into database tables. There are also housekeeping scripts called `filexfer-filearchive`, which keeps files in the data directory pruned and compressed, and `filexfer-fileunarchive`, which allows files to be pulled out of the archive so filexfer jobs can work with them again."[1]

---

[1]Usage Collection Framework (filexfer)

Contents

List of Figures

List of Tables

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

List of Definitions and Abbreviations

- **MOA** - Municipality of Anchorage

1 Introduction

The FileXfer system...

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

2   Design

[FIXME: Need data here...]
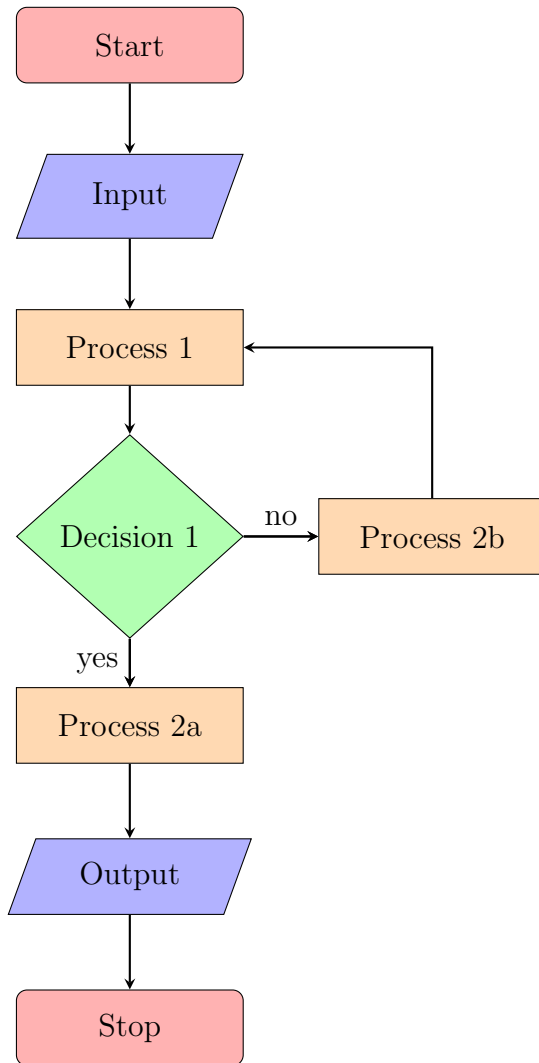
[FIXME: Customize flowchart FileXfer]

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

3   Implementation

[FIXME: Need data here...]

3.1   Logging

Application Logging

The filexfer applications log to the `/var/log/filexfer` directory on `prod-prov4-cdr1.\ operations.gci.com`. The parent `filexfer` jobs log to `filexfer-get.log` and `filexfer-\ put.log`. The `jobmonitor` and `dataloader` applications log to `jobmonitor.log` and `dataloader.log`, The `filexfer` applications log to the `/var/log/filexfer` directory on `prod-prov4-cdr1.operations.gci.com`. The parent `filexfer` jobs log to `filexfer-get.\ log` and `filexfer-put.log`. The `jobmonitor` and `dataloader` applications log to `jobmonitor.log` and `dataloader.log`, respectively. Each file transfer job is executed as a child process and gets its own log file. The format is
`filexfer-{neName}-{idJob}-{get,put}.log`.

By default, the jobs log at the warn level. Adjust the level to info to get a high-level view of the application's state. Adjust log verbosity by modifying the appropriate config file in `/etc/filexfer`. The changes will take effect after the next program execution.

Errors are also logged to a database table which can be browsed in the filexfer web interface under the 'Logs & Errors' view. This view includes messages logged at `warn`, `error`, and `fatal` severity.[2]

File Transfer Logging

Every file transfer is recorded in a database table. There are two reasons for this table: first, it tells `filexfer` hich files have already been transferred, and second, it provides an audit trail for SOX compliance. The table is `filexfer.logs` on `sadc-cdr-mysql1.operations.gci.com`. Use the `filexfer.joblogs` view to easily find logs by job name or network element ID.

File transfer logs may also be viewed in the 'Logs & Errors' page of the web interface.[3]

---

[2]Usage Collection Framework (filexfer)
[3]Usage Collection Framework (filexfer)

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

4   Test

[FIXME: Need data here...]

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

5 Issues

[FIXME: Need data here...]

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

6 Operation

[FIXME: Need data here...]

6.1 Job Scheduling

Jobs are scheduled using a web interface at `nms.operations.gci.com/relevance`. Navigate to the "FileXfer" application and click the "File Transfer Jobs" link. Job execution happens on `prod-prov4-cdr1.operations.gci.com`. A `cron` job executes every minute from `/etc/cron.d/filexfer` to kick off the various `filexfer` scripts.[4]

Job Timing

The parent `filexfer` script is responsible for spawning child processes for each job. Since a large number of jobs can be scheduled at any given interval, the parent process limits how many children can run concurrently. As long as the limit is reached and more jobs need to be spawned, the parent process must stay alive. Since this may take longer than 1 minute, it is possible for `filexfer` to miss certain scheduling intervals.

For example, if 500 jobs are scheduled to run at the top of every hour (`0 * * * *`) and the maximum child process limit is 50, there is a good chance `filexfer` will not execute any jobs scheduled to run at 1 minute past the hour (`1 * * * *`). The best way to avoid this is to use `0`, `15`, `30`, or `45` in the minute field of the job schedule. These intervals are always executed.[5]

6.2 Dataloader

Dataloader jobs are configured using the web interface at `nms.operations.gci.com/relevance`. Navigate to the "`FileXfer`" application and click the "`Data Load Jobs`" link. These jobs are executed every minute as long as there are files in the load queue.[6]

---

[4]Usage Collection Framework (filexfer)
[5]Job Timing
[6]Usage Collection Framework (filexfer)

Created June 30, 2016 from `filexfer.tex` (sha-1: `2c274da`)   10
by Raymond E. Marcil <`marcilr@gmail.com`>

7 Examples

Series of useful LaTeX markup. Need to break out to separate examples.tex file.

7.1 Escaping $<$ and $>$ Symbols

To get $<$ or $>$ just wrap the symbols in $ for math mode.

7.2 Enumerate

1. DNR - Alaska State Department of Natural Resources

   - HI - Historical Index, not maintained since 1982
   - LE - Land Estate, maintained by SGU
   - ME - Mineral Estate, maintaind by SGU

2. Alaska State Surveys

   - ASBLT - As-Built Survey
   - ASCS - Cadastral Survey

7.3 Comments

> COMMENTS Comment — *Sean Weems, Spring 2003*
> We should get the COMMENTS column searchable via the landrecords application before we do much anything else – shouldn't be too hard.

> *Errata: Plats spanning multiple sections*
> A few anomalies can be observed in the AKPLATS table. Specifically plats exist that span multiple sections. Since the table only has a single column, SCODE, that accepts a single section code, SGU (Status Graphics Unit) has handled this problem by entering multiple rows in the table, each with a different section that point to the same plat or file. Multiple section plats are indicated by setting the TCODE column to the value 37, and making an appropriate notation like *Section 24-25-26-27* in the REMARKS column.
> [FIXME: Perhaps the SCODE column should accept an array of sections?]

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

7.4   Footnotes

See my footnote[7] generated with:

```
\footnote{\href{http://www.google.com/search?q=latex+footnotes}
{Search google for footnotes.}}
```

GoogleGuide — Linking to Search Results.[8]

7.5   Hyperlinks

Use \href{} to generate hyperlinks:

```
\href{http://www.google.com}{Google}}
```

Yields: Google

7.6   Table Examples

Column Name	Type	Description
EQS	VARCHAR2(1)	!NULL map shows village selections
ITM_COL	VARCHAR2(1)	USGS ITM column: 1-6
ITM_ROW	VARCHAR2(1)	USGS ITM row: A-E
QMQ_ABBR_DNR	VARCHAR2(3)	Three character DNR abbreviation for the QMQ
RASTER_FILENAME	VARCHAR2(50)	Physical path to file
RASTER_PATHNAME	VARCHAR2(50)	URL path to PDF of map
SCODE	VARCHAR2(2)	Supplement map code: 1,2,3,...
COMMENTS	VARCHAR2(256)	Plat comments

Table 1: `EASEMENTS_17B` Table

---

[7]Search google for footnotes.

[8]GoogleGuide — Linking to Search Results.

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

XML element	Descripton
FNUM	US Survey file number
MERIDIAN	BLM meridian code
	12 = Copper River
	13 = Fairbanks
	28 = Seward
	44 = Kateel
	45 = Umiat
TOWNSHIP	Five character Township code
RANGE	Five character Range code
PAGE	Survey page number 1,2,3,...
FILENAME	Relative path to file in direcory

Table 2: USS XML index elements

col 1	col 2	col 3	col 4
item 1	item 2	item 3	item 4
item 1	item 2	item 3	item 4

Table 3: Demo

Virtual Machine	Apache	ELM	LM	Elluminate Server
dcs-elive-prod01		x	x	x
uaa-elive-dev01	x	x	x	
uaa-elive-server01				x
uaa-elive-prod01		x	x	x
uaf-elive-prod01		x	x	x
uas-elive-prod01		x	x	x

Table 4: Daemons

Column Name	Type	Description
MTR	VARCHAR2(9)	Meridian, Township, Range, example: *C026S054E*
QMQ	VARCHAR2(3)	Quarter Million Quadrangle code,
		example: *DIL* (Dillingham quadrangle)

Table 5: `XREF_MTR_QMQ` Table

7.7   Verbatim

"The verbatim environment is a paragraph-making environment that gets LaTeX to print exactly what you type in. It turns LaTeX into a typewriter with carriage returns and blanks having the same effect that they would on a typewriter."[9]

```
\begin{verbatim}
    text
\end{verbatim\}
```

Figure formatting with verbatim
The following figure leverages verbatim for proper formatting:

```
gis/raster/
  dnr/
    map_library/
    plats/
      SP/YYYYMMDD/*.pdf              # indexed
      HI/YYYYMMDD/*.pdf              # Indexed
      ASLS/YYYYMMDD/*.pdf           # Indexed
    recorded-plats/
      YYYYMMDD/*.pdf
  blm/
    easements_17b/YYYYMMDD/*.pdf    # indexed
    mtp/YYYYMMDD/*.pdf              # non-indexed
    usrs/YYYYMMDD/*.pdf             # indexed
    usrs-notes/YYYYMMDD/*.pdf       # indexed
    uss/YYYYMMDD/*.pdf              # indexed
    uss-notes/YYYYMMDD/*.pdf        # indexed
    usms/YYYYMMDD/*.pdf             # indexed
    usms-notes/YYYYMMDD/*.pdf       # indexed
  usgs/
    drg/
      collared/
        250K/
        63K/
        25K/
        24/
      decollared/
      tools/
      missing\_data/
    dem/
    doq/
    topo/
```

Figure 1: File and Directory Structure

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

Appendix

Source

There are 3 primary FileXfer `perl` scripts on `prod-prov4-cdr1`:[10]

File name	Attributes	Description
`filexfer.plx`	181 lines	File transfer jobs
`filexfer-dataloader.plx`	132 lines	Data loader
`filexfer-jobmonitor.plx`	132 lines	Job Monitor

Table 6: FileXfer `perl` scripts on `prod-prov4-cdr1`

---

[10]`prod-prov4-cdr1.operations.gci.com` (192.168.161.47, NATed IP: 66.223.199.228), data including CDRs and such under `/data/usage/` — Network Services, OSS.

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

filexfer.plx — File Transfer Jobs

```perl
#!/usr/bin/perl

use strict;
use warnings;

use Modules::App::FileXfer ();
our $VERSION = $Modules::App::FileXfer::VERSION;

# Core modules
use Clone qw( clone );
use File::Basename ();
use File::Spec ();
use POSIX ();

$SIG{CHLD} = \&Modules::App::FileXfer::REAPER;

MAIN: {
    # Process and merge command-line and config file options
    my $getopt   = Modules::App::FileXfer::get_command_line_options();
    my $fileconf = Modules::App::FileXfer::read_config_file( $getopt->get_configfile );
    Modules::App::FileXfer::merge_options( $getopt, $fileconf );

    # Make sure we're the only instance running
    Modules::App::FileXfer::check_pid_file( $Modules::App::FileXfer::Options->{pidfile}

    # Get logger and evenge objects
    Modules::App::FileXfer::create_evenge_obj();
    my $logger = Modules::App::FileXfer::create_logger_obj(
        $Modules::App::FileXfer::Options->{logger}, $Modules::App::FileXfer::Program );

    # Get the ready jobs
    my $fx       = Modules::App::FileXfer::create_filexfer_obj( $Modules::App::FileXfer:
    my $jobs     = Modules::App::FileXfer::get_jobs( $fx );
    my $loadjobs = Modules::App::FileXfer::get_jobs_with_load_jobs( $fx );
    undef $fx;

    for my $job ( @{ $jobs } )
    {
        # Enforce the "max children" constraint
        $logger->info( 'Max child processes reached. Waiting for one to complete before
            if ( scalar keys %Modules::App::FileXfer::Children
                >= $Modules::App::FileXfer::Options->{maxchildren} );
```

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

```
        sleep 1 while ( scalar keys %Modules::App::FileXfer::Children
                        >= $Modules::App::FileXfer::Options->{maxchildren} );

    # Fork a child process for this job
    $logger->info( sprintf( 'Spawning child process for job "%s".', $job->jobName ))

    my $pid = fork;
    defined $pid or Modules::App::FileXfer::log_event(
        5, sprintf( "Can't fork for job \"%s\": %s", $job->jobName, $! ), 'logdie' )

    if ( $pid == 0 ) # child
    {
        # Set random seed for this child
        srand();

        # Lower the OS scheduling priority based on job priority
        POSIX::nice( Modules::App::FileXfer::pri_to_nice( $job->priority ));

        # Add the NE name to our command line string
        $0 .= " @ARGV " . $job->neName;
        my $jobtag = Modules::App::FileXfer::get_jobtag( $job );

        # Set the subresource for this job in the evenge object
        $Modules::App::FileXfer::Evenge->subresourceName( $job->jobName );

        # Create a logger specific to this child process
        my $logopt = clone( $Modules::App::FileXfer::Options->{logger} );
        my ( undef, $logdir ) = File::Basename::fileparse( $logopt->{file}{filename}
        $logopt->{file}{filename} = File::Spec->catfile( $logdir, "$jobtag.log" );

        $logger->delete();
        my $logger = Modules::App::FileXfer::create_logger_obj( $logopt, $job->jobNa
        Log::Log4perl::MDC->put( 'idJob', $job->idJob );

        # Make sure another instance isn't still running
        Modules::App::FileXfer::check_pid_file( $jobtag );

        # Create a FileXfer object for database updates
        my $fx = Modules::App::FileXfer::create_filexfer_obj( $jobtag );

        # Execute the job
        Modules::App::FileXfer::run_job( $fx, $job, $loadjobs );
```

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

```
            $logger->info( 'Child exiting.' );
            exit 0;
        }
        else # parent
        {
            $logger->debug( sprintf( 'Spawned child process %d for job "%s".', $pid, $jo
            $Modules::App::FileXfer::Children{ $pid } = $job->jobName;
        }
    }

    $logger->info( 'Main application exiting.' );
}

# Safely exit
$SIG{CHLD} = 'IGNORE';

__END__

=head1 NAME

filexfer -- Move a file from point A to point B over an IP network

=head1 VERSION

0.51

=head1 SYNOPSIS

filexfer.plx -c configfile -t {get|put} [options]

=head1 ARGUMENTS

=over 4

=item -c, --configfile

Specify the configuration file to load. Must be in YAML format.

=item -t, --transfertype

One of "get" or "put". Get jobs download files and put jobs upload files.

=back
```

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

=head1 OPTIONS

=over 4

=item -d, --piddir

Directory where the pid file will be written. Defaults to /var/run/filexfer.

=item --db

Sets the database connection parameters. Valid keys are: server (default
localhost), port (default 3306), driver (default mysql), uid, pwd, database,
and table. Specify tags as key/value pairs, e.g.:

    --db server=localhost --db database=filexfer

=item -e, --evengehost

Address of the Evenge web server. Used to send indicators and events to the NMS system.

=item --evengetimeout

Timeout in seconds for communicating with the Evenge web server. Defaults to 10.

=item -f, --cachefile

Template cache file location. Defaults to /var/lib/filexfer/filexfer.kch.

=item -h, --help

Output this documentation.

=item -m, --maxchildren

Maximum number of child processes to spawn. Defaults to 50.

=item -p, --pidfile

PID file name. This will be appended with a ".pid" suffix.

=item -r, --resource

Resource name of this application. Used in indicator and event messages sent to the NMS

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

```
=item --verbose, -v
```

Log to the screen at increasingly verbose levels. This option may be repeated
multiple times to increase the log level. For example, "-v" logs at info level,
"-vv" logs at debug level, and "-vvv" logs at trace level.

```
=back
```

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

filexfer-jobmonitor.plx — Job monitor

```perl
#!/usr/bin/perl

use strict;
use warnings;

use Modules::App::FileXfer::JobMonitor ();
our $VERSION = $Modules::App::FileXfer::JobMonitor::VERSION;

MAIN: {
    # Process and merge command-line and config file options
    my $getopt   = Modules::App::FileXfer::JobMonitor::get_command_line_options();
    my $fileconf = Modules::App::FileXfer::JobMonitor::read_config_file( $getopt->get_co
    Modules::App::FileXfer::JobMonitor::merge_options( $getopt, $fileconf );

    # Make sure we're the only instance running
    Modules::App::FileXfer::JobMonitor::check_pid_file(
        $Modules::App::FileXfer::JobMonitor::Options->{pidfile} );

    # Get logger and evenge objects
    Modules::App::FileXfer::JobMonitor::create_evenge_obj();
    my $log = Modules::App::FileXfer::JobMonitor::create_logger_obj(
        $Modules::App::FileXfer::JobMonitor::Options->{logger},
        $Modules::App::FileXfer::JobMonitor::Program
    );

    # Get a list of job monitors
    my $jm = Modules::App::FileXfer::JobMonitor::create_jobmonitor_obj();
    my $mons = Modules::App::FileXfer::JobMonitor::get_monitors( $jm );

    for my $mon ( @{ $mons } )
    {
        Log::Log4perl::MDC->put( 'idJob', $mon->idJob );
        $log->info( sprintf( 'Executing monitor "%s".', $mon->monitorName ));

        # Set the subresource for this monitor in the evenge object
        $Modules::App::FileXfer::JobMonitor::Evenge->subresourceName( $mon->monitorName

        # Execute the job monitor
        eval { Modules::App::FileXfer::JobMonitor::run_monitor( $jm, $mon ) };
        $@ and $log->error( "$@" );
    }
```

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

```
    $log->info( 'Main application exiting.' );
}
```

`__END__`

`=head1 NAME`

`jobmonitor -- Monitor filexfer jobs for any condition and generate alerts`

`=head1 VERSION`

`0.51`

`=head1 SYNOPSIS`

`jobmonitor.plx -c configfile [options]`

`=head1 ARGUMENTS`

`=over 4`

`=item -c, --configfile`

`Specify the configuration file to load. Must be in YAML format.`

`=back`

`=head1 OPTIONS`

`=over 4`

`=item -a, --mailhost`

`Address of the mail server. Used to send email notifications. Defaults to localhost.`

`=item -d, --piddir`

`Directory where the pid file will be written. Defaults to /var/run/filexfer.`

`=item --db`

`Sets the database connection parameters. Valid keys are: server (default localhost), port (default 3306), driver (default mysql), uid, pwd, database, and table. Specify tags as key/value pairs, e.g.:`

```
    --db server=localhost --db database=filexfer
```

=item -e, --evengehost

Address of the Evenge web server. Used to send indicators and events to the NMS system.

=item -f, --cachefile

Template cache file location. Defaults to /var/lib/filexfer/jobmonitor.kch.

=item -h, --help

Output this documentation.

=item -i, --mailinterval

Minimum time, in seconds, before repeat emails may be sent for the same monitor. Default

=item -l, --mailfrom

Sender's email address in email notifications. Defaults to [username]@[hostname].

=item -m, --mailstatfile

Mail status file location. Defaults to /var/lib/filexfer/jobmonitor-mailstat.kch.

=item -o, --mailport

Port on which the mail server is listening for SMTP traffic. Defaults to 25.

=item -p, --pidfile

PID file name. This will be appended with a ".pid" suffix.

=item -r, --resource

Resource name of this application. Used in indicator and event messages sent to the NMS

=item --verbose, -v

Log to the screen at increasingly verbose levels. This option may be repeated
multiple times to increase the log level. For example, "-v" logs at info level,
"-vv" logs at debug level, and "-vvv" logs at trace level.

=back

[FIXME: Need data here]

Table 7 – FileXfer directories and files on `prod-prov4-cdr1`

Directory	File(s)
/etc/filexfer/	*.conf
/usr/bin/	filexfer-dataloader
	filexfer-dataloader.plx
	filexfer-dataloader.plx.mbak
	filexfer-epg-dataloader.plx
	filexfer-filearchive
	filexfer-filearchive.sh
	filexfer-fileunarchive
	filexfer-fileunarchive.sh
	filexfer-jobmonitor
	filexfer-jobmonitor.plx
	filexfer.plx
/usr/lib/filexfer/	*.gz, *.sh, *.plx
	ExtractCarrierTurboZoneUsage*
/usr/share/filexfer/	filexfer.changelog-*.xml
	filexfer.changelog-master.xml
	liquibase.sh
	.gnupg/pubring.gpg
	.gnupg/random_seed
/var/cache/yum/build/packages/	filexfer-0.52-1.el5.centos.noarch.rpm
/var/lib/filexfer/	dataloader.kch
	dataloader_temp.kch
	filexfer-aaa01-13-get.kch
	...
	filexfer-wps01-706-get.kch
	filexfer.kch
	jobmonitor-mailstat.kch
	jobmonitor.kch
/var/log/filexfer	*.log
	ExtractCarrierTurboZoneUsage_ACS.log
	archive
	convert-wps-om-counters-report-part2.log
	convert-wps-om-counters-report.log
	dataloader.log
	dataloader_temp.log

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --

Table 7 – continued from previous page

Directory	File(s)
	`dataloadinsert.log`
	`datarecovery.log`
	`epg-dataloader.log`
	`ericsson-oss-rl-reports-preprocess.log`
	`ericsson-oss-sts-reports-preprocess.log`
	`filearchive.log`
	`filexfer-aaa01-13-get.log`
	`filexfer-aaa01-14-put.log`
	`...`
	`/var/log/filexfer/filexfer-wps01-706-get.log`

Links

A Guide to LaTeX
http://www.astro.rug.nl/ kuijken/latex.html

LaTeX - From Wikibooks,the open-content textbooks collection
http://en.wikibooks.org/wiki/LaTeX

LaTeX Notes
http://luke.breuer.com/time/item/LaTeX_Notes/180.aspx

DRAFT -- DRAFT -- DRAFT -- DRAFT -- DRAFT --