NASA Applied Sciences
DEVELOP National Program

NASA Langley Research Center

August 8, 2011

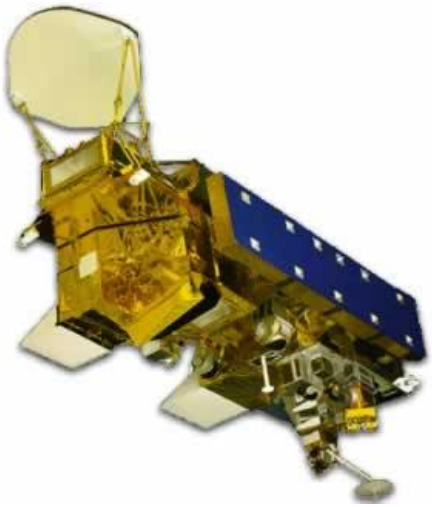
Project Overview

- Perl_Lib comments for all subroutines and modules, as found in CERESLib
- Documentation Reference Generator
- Detailed documentation for Perl_Lib
- Dates module changes

Perl_Lib

Modules

- Dates
- PathUtils
- SGE_Utils
- PCF_Env
- Archive_Utils
- Pcf
- Summary



Perl_Lib

- **Dates**
  - Date Format Checking
    - Check_Date, Check_Date_Day, Check_Date_Month, Check_Year, Check_Month, Check_Hour, Check_Day
  - Date Math
    - Get_Next_Date_Hour, Get_Next_Date_Day, Get_Next_Date_Month, Get_Next_Date
    - Get_Prev_Date_Hour, Get_Prev_Date_Hour, Get_Prev_Date_Month
    - Increment_Day, Increment_Hour, Increment_Month
  - Date Formatting
    - Split_Date_Hour, Split_Date_Day, Split_Date_Month, Split_Hyphen_Date_Day, Split_Hypen_Date_Hour, Split_Hyphen_Date_Month
    - Insert_Hyphens_Date_Day, Insert_Hyphens_Date_Month, Insert_Hyphens_Date_Hour
    - Remove_Hyphens_Hour, Remove_Hyphens_Day, Remove_Hyphens_Month
  - Misc
    - Is_Leap_Year, Get_Julian_Day, Day_Of_Year, Get_Days_In_Month, Verify_Date_Selection

Perl_Lib

- **Pcf**
  - Writing PCF files
    - To_Pcf, To_Pcf_Met, Print_Header, Print_Footer, Print_Seperators, Print_Comments, Print_TK_Files, Print_Log_Files, Print_User_Param, Print_Users_Vars, Print_Meta_Files.
  - Logging
    - Print_Log

- **Archive_Utils**
  - Path and file name utilities
    - Get_Directory
    - Add_Daily, Add_Daily_Met, Add_Hourly, Add_Hourly_Met, Add_Zonal, Add_Zonal_Met, Add_Monthly, Add_Monthly_Met

- **SGE_Utils**
  - Sun Grid Engine job control
    - Submit, getCPUTYPE, sgeLogHeader, CheckFileLog, MatchPlatform, setupEnvironment, CheckPlatform, CheckForExecutables
- **PCF_Env**
  - Vault hash creation
    - setUpPcfHash
- **Summary**
  - Log file creation and extraction
- **PathUtils**
  - Path functions
    - parentDir

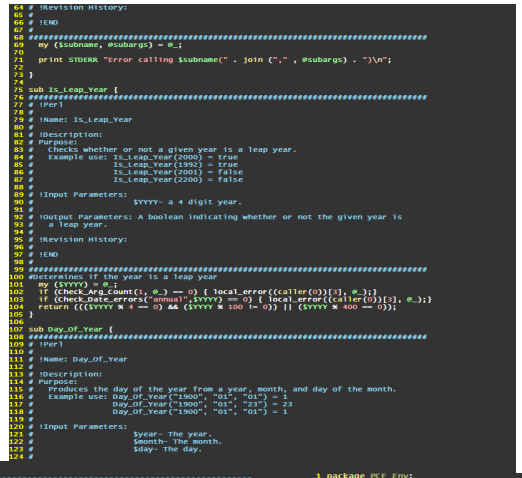
CERES Prologue Extraction

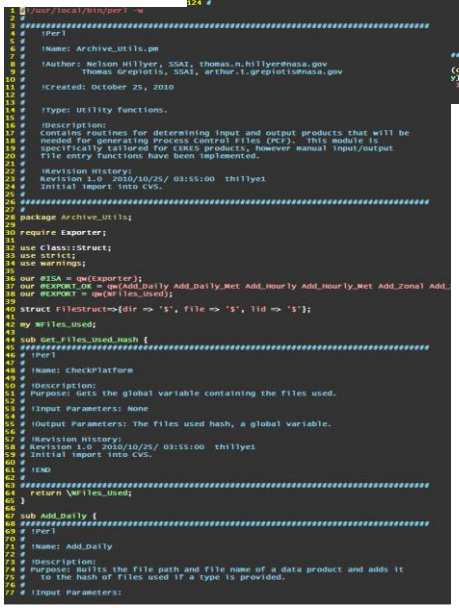
- **Command line tool**
- **Generates a static web page from source code comments**
- **Written for Perl_Lib, but will extract any comments in the standard format given the comment character used**
- **Example use**
  - ceresprologue.pl
  - ceresprologue.pl Dates.pm
  - ceresprologue.pl . –e “f90|csh”
  - ceresprologue.pl codedir –n CERESlib –d docdir

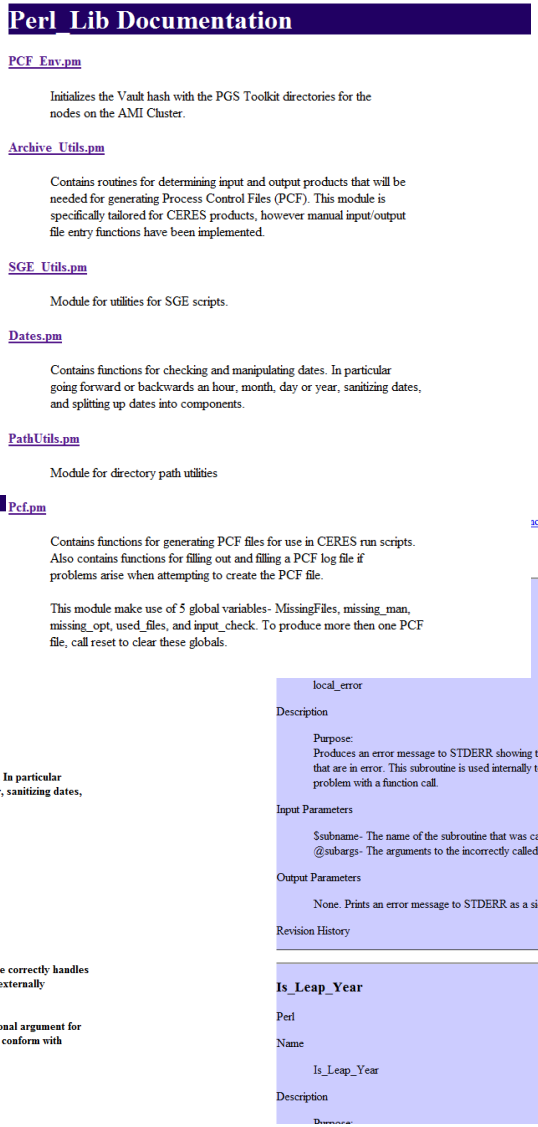
```
      $content =~ s/Name://g;
      }
      push(@block, $content);
    }
    push(@docs, \@block);
  }
  return \@docs;
}

sub cleanHeader {
  my $header = shift;
  trim($header);
}

sub cleanContent {
  my $content = shift;
  $content =~ s/\s*$cmt\s*/\n/g if defined $content;
  $content = trim($content);
  $content =~ s/\n/<br>/g if defined $content;
  $content;
}

sub trim {
  my $str = shift;
  $str =~ s/(^\s*|\s*$)//g if defined $str;
  return $str;
}

sub printComments {
  my $doc = shift;
  print "length: " . scalar @$doc;
  foreach my $assoc (@$doc) {
    foreach (1..80){print "+"};
    print concatMapAssoc(sub { "section: $_[0]\ncontains: $_[1]\n" }, $assoc);
  }
}

sub headerHTML {
  return start_html(-title=>$projectName,
    -style=>{"src"=>'./style.css'});
}

sub toHTML {
  my $doc = shift;
  my $str = headerHTML;
  $str .= h3({class=>"title"}, $projectName);
  $str .= concatMapAssoc(sub {h3($_[0]) . h4(indent($_[1]))}, shift @$doc);
  $str .= concatMap(\&anchor, $doc, ", ");
  $str .= h2("Subroutines: ") . concatMap(\&methodHTML, $doc);
  $str .= end_html;
}

sub anchor {
  my $assoc = shift;
  my $nam = lookup("Name", $assoc);
  a({href=>"#$nam" }, $nam);
}

sub concatMap {
  my ($sub, $list, $sep) = @_;
  $sep = "" unless defined $sep;
  join($sep, map({&$sub($_)} @$list));
}
```

```
##############################################################################
# !Perl
#
# !Name: CheckPlatform
#
# !Description
# Purpose: Checks a PGE hash's platform entry against a given platform.
#
# !Input Parameters:
#                    $pge_hash_ref- A reference to a hash containing enviroment
#                     information.
#                    $platform- The platform to check against.
#
# !Output Parameters: A boolean indicating whether or not the given platform
#                       matches the platform entry in the provided hash.
#
# !Revision History:
#
# !END
#
##############################################################################
```

Documentation Reference







Detailed Documentation

- **Based off of ERBE documentation**
- **Control flow diagrams**
- **Pseudo-code explanations of subroutine implementation**

Log file for: Pcf_File

*****Input Variables used:*****

KEY => VALUE from array referenced by CERES_Vars_Ref

****PGE variables used:*****

KEY => VALUE from array referenced by PGE_Vars

Output from InputFileInfo, one array element per line.

Where this logging information is written to the given File_Handle file handle.

Split_Filename- Splits the given file path into a path name paired with the file name.

Example Use: Split_FileName("/dev/null") = ("/dev", "null")

AddtoMissingMan- For logging manditory files only. Does not write anything to a PCF file. The only argument is the file name, which is added to the missing_man array and MissingFiles in the "Mandatory" entry's array.

AddtoMissingOpt- For logging optional missing files only. Does not write anything to a PCF file. The only argument is the file name, which is added to missing_opt and MissingFiles in "Optional" entry's array.

AddtoUsed- Adds the given file name to the global array of used files, used_files.

InputFileInfo- Produces an array of strings containing the names of the files in the global file lists. This information is in the following form:

*****Files Used:*****

Used files from used_files array, or --NONE--

*****TYPE Files Missing:*****

File names from TYPE entry in Missing_Files hash, or --NONE--

Where TYPE is the type of the files, with Mandatory and Optional always included as types, and possibly others as keys in the MissingFiles hash.

Reset- Resets the global variables for missing files to empty. This subroutine must be called between each PCF generated. The globals that are reset are MissingFiles, used_files, missing_man, and missing_opt.

if platform contains p6, platform = ppc64

if platform contains x86, platform = x86_64

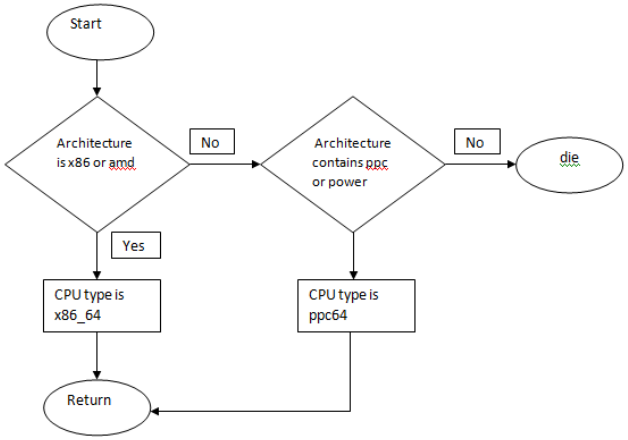
}

If hash has no "executables" entry then die

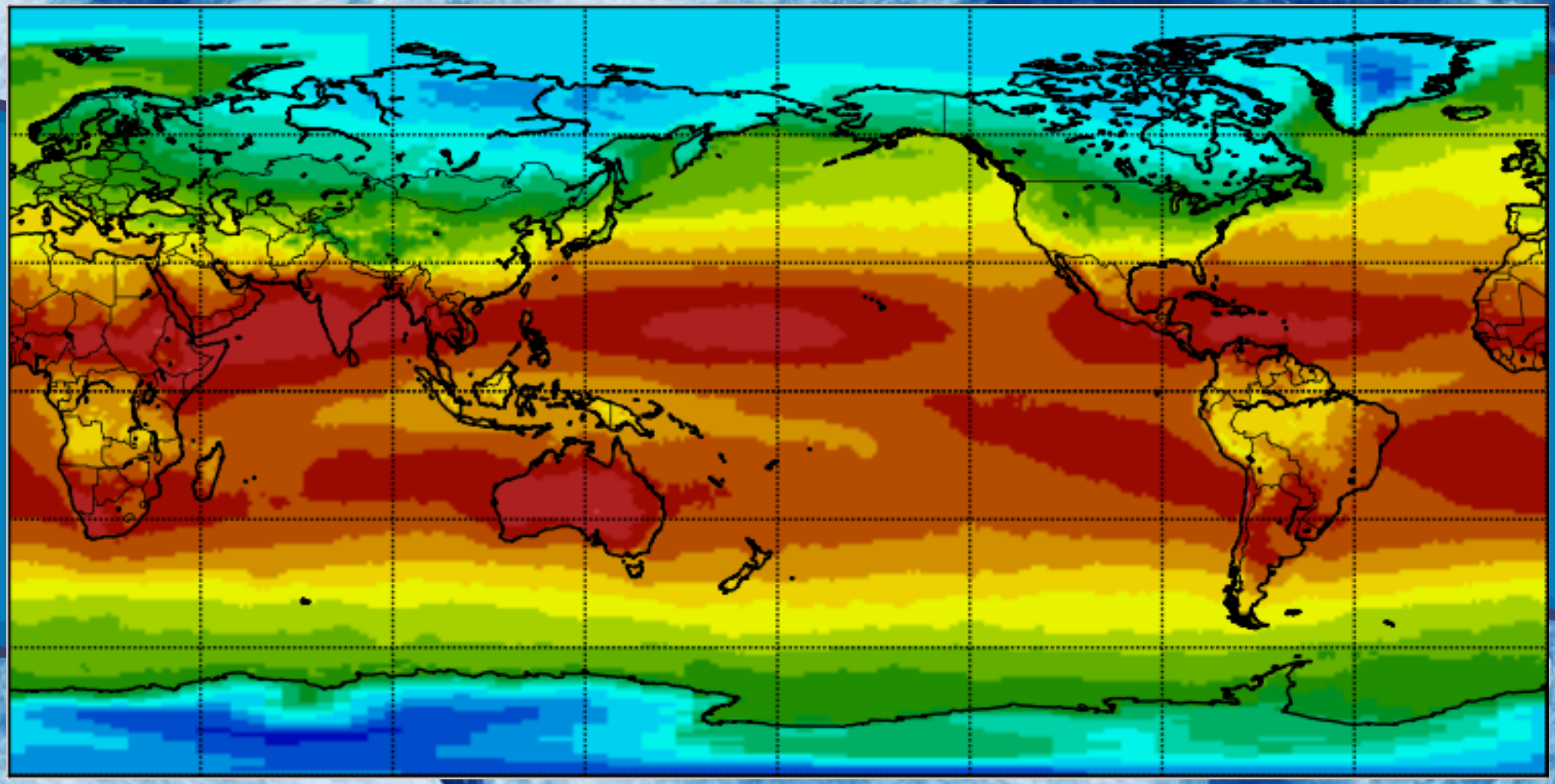
return executables from hash's executable entry that are empty, do not exist, or does not contain the platform name (if it is given).

}

getCPUType- This subroutine relies on the "arch" command line program to determine the architecture of the platform. The output of the arch command is sanitized to read either X86_64 or ppc64. If the output does not match x86_64, amd64, power, or ppc then the subroutine exits with an error.



setupEnvironment- The environment may require some additional commands to be run before it is ready to be used for processing tasks. This subroutine runs these commands. In particular it sets the umask such that the file/directory owners and group have read and write permissions to files written through the SGE interface.

Changes to Dates.pm

- **Time::Piece From Perl's standard library**
- **Changed-**
  - Day_Of_Year, Get_Prev_Date_Hour, Get_Next_Date_Hour, Get_Prev_Date_Day, Get_Next_Date_Day, Get_Julian_Day, Split_Date, Split_Date_Hour, Split_Date_Day, Split_Date_Month, Split_Hyphen_Date_Day, Split_Hyphen_Date_Month, Split_Hyphen_Date_Hour, Insert_Hyphens_Date_Day, Insert_Hyphens_Date_Month, Insert_Hyphens_Date_Hour, Remove_Hyphens_Day, Remove_Hyphens_Month, Remove_Hyphens_Hour.