Update of the CERSER TeraScan Cataloguing System and the TeraScan Image Processing Scripts

Derek Morris Jr., Jefferson Ridgeway, Tori Wilbon, Andrew Brumfield
Center of Excellence in Remote Sensing Education and Research
Elizabeth City State University
Elizabeth City, North Carolina, USA

Abstract - **The Center of Excellence in Remote Sensing Education and Research (CERSER) on the campus of Elizabeth City State University is currently tasked with the responsibility of receiving remotely sensed data from National Oceanic and Atmospheric Administration (NOAA) Polar Operational Environmental Satellites (POES) and Geostationary Operational Environmental Satellites (GOES). This data is collected by SeaSpace TeraScan systems installed in the CERSER labs in Dixon-Patterson Hall.**

In 2005, the processing system underwent a major update due to a migration to a new operating system. A minor update was needed at this time to deal with a second operating system migration and display of the processed images on the CERSER web site. Since then, a second transfer to a new server was made in 2013. The cataloguing system went down at this time and was not repaired due to technical issues with the TeraScan system. The 2014 team corrected issues within the current server directory system and updated the data script to process images from the GOES-13 satellite received by the TeraScan system. Software and languages utilized for this task included ImageMagick, PHP, HTML, Dreamweaver, phpMyAdmin, and MySQL.

Along with this operating system update, a major script development was needed on the TeraScan processing equipment due to an upgrade in hardware. The ground station upgrades included a 3.7m X/L band, a 3.6m C band, and a 5.0m L band dishes, along with accompanying computing hardware.

This project implemented new script on the TeraScan and CERSER servers. This new script processes both infrared and visible light images received from the GOES-13 satellite into the Tagged Image File (TIFF) Format.

Key Words - **GOES, TeraScan, ImageMagick, Macintosh OSX, MySQL, PHP, Remote Sensing**

I. INTRODUCTION

In 2013, the Center of Excellence in Remote Sensing Education and Research (CERSER) added more satellite receiving hardware to its current National Oceanic and Atmospheric Administration (NOAA) satellite system [2]. The program now processes both NOAA satellite images as well as Geostationary Operational Environmental Satellite (GOES) [8] satellite images. This new imaging capability required the development of scripts on the TeraScan system to process satellite data received and produce an image file with overlay features such as latitude/longitude lines and political map outlines.

During this expansion, the image processing files had also been migrated to another server. This migration resulted in the failure of the previously developed scripts tasked with the processing of the Tagged Image File Format (TIFF) satellite images provided by the TeraScan system. This failure had not been repaired due to the mechanical failure of the previous satellite receiving system. These scripts parsed the title of the image file into database records and transformed and converted the TIFF image into various Joint Photographic Experts Group (JPEG) files. The scripting on the server side that was utilized and reviewed was the code from the 2011 research team and is written in PHP Hypertext Processor (PHP) and My Structured Query Language (MySQL) [10].

The first goal of this project was to create the scripts needed on the TeraScan system in order to process the GOES satellite data. The second objective was to reconfigure the scripts created by the 2011 team to repair and update the server processing capability to process the new GOES images into the current image database and file system on the CERSER server.

II. SOFTWARE AND LANGUAGE

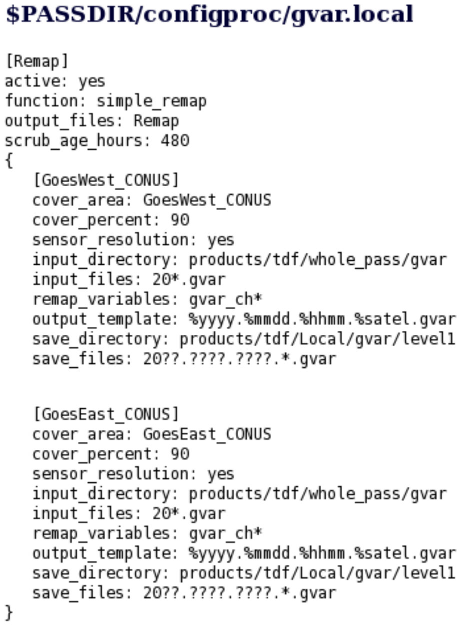
A. *TeraScan*

TeraScan is a combination of hardware and software designed for automated reception of data from meteorological/environmental satellites and for processing the data into images and data overlays [1]. Images and overlays are of a special format called TeraScan Data Format (TDF) and can be displayed with the TeraScan viewer called TeraVision. These data files can be used to produce images in various formats. Different types of satellite data can be received from both the hardware and software that is installed on the computer. These satellite data types include: X-Band Data from Polar-Orbiting Satellites, L-/S- Band Data from Polar-Orbiting Satellites, and L-Band Data from Geostationary Satellites [1].

There are several different structures of hardware for the TeraScan system in how it receives and processes data. In the system, there is an antenna, receiver, bit synchronizer, frame synchronizer, computer workstation with TeraScan software, Global Positioning System (GPS), and an uninterruptible power supply (UPS) [1]. Different antennas are used for different satellite signal interception. For high-resolution data reception from geostationary satellites, a fixed-direction antenna is used. For high-resolution data reception from polar-orbiting satellites, a tracking antenna is used. The receiver is used for tuning to the correct satellite signal frequency and separating the sensor data (the baseband) from the RF carrier (the latter function is known as demodulation). The bit synchronizer regenerates the bit stream against noise, and if necessary, extracts the clock, which is a timing pulse added to the signal on the satellite to enable synchronization of the reception system with the satellite transmitter. The frame synchronizer divides the continuous bit stream into meaningful data blocks called frames [1]. The computer workstation helps control data reception and process the data into products. Finally, the GPS supplies position and time to the system and the UPS protects the system from power surges [1].

SeaSpace supports TeraScan on two software platforms for its system: Sun Solaris and RedHat Linux. These software platforms are based on the UNIX operating system and consist of more than 500 command-line functions, daemons, reference files, a set of graphical user interfaces (GUIs), and TDFs [1]. Two of the GUIs that were used were TeraVision and TeraMaster. TeraVision was used for displaying and editing TeraScan data images and overlays. Some of the capabilities of TeraVision include the image enhancement via color palettes, convolution filters, histogram equalization, manipulation of the view area by panning and zooming, and the animation of a series of images (time/product looping) [3]. TeraMaster was used for viewing, creating, and modifying masters. A master, also known as an area of interest (AOI), a TeraScan dataset that defines an area of the earth’s surface in terms of map projection (shape), extents, and pixel resolution. A master has three purposes in TeraScan: 1. Sets criteria for the schedule of satellite passes, 2. Establishing a smaller set of data from a whole pass for processing, and 3. A base map view is changed to an earth-located view of the data which is known as registration [3].

Data processing is carried out according to configuration files located in $PASSDIR/configproc. The processes to be run are set as "active" in those files [6]. Batch.ingest files within the configproc directory specifies for ingest function(s) to be run by run_ingest, including gvarin, which makes whole pass gvar datasets. Gvar.local specifies remapping of the whole pass datasets to the Local Master [6]. Gvarin reads GOES VARiable Format (GVAR) data directly from application shared memory when reception is live [9]. GVAR is the data transmission format used with the new generation of GOES meteorological satellites [10].



```
$PASSDIR/configproc/gvar.local

[Remap]
active: yes
function: simple_remap
output_files: Remap
scrub_age_hours: 480
{
   [GoesWest_CONUS]
   cover_area: GoesWest_CONUS
   cover_percent: 90
   sensor_resolution: yes
   input_directory: products/tdf/whole_pass/gvar
   input_files: 20*.gvar
   remap_variables: gvar_ch*
   output_template: %yyyy.%mmdd.%hhmm.%satel.gvar
   save_directory: products/tdf/Local/gvar/level1
   save_files: 20??.????.????.*.gvar


   [GoesEast_CONUS]
   cover_area: GoesEast_CONUS
   cover_percent: 90
   sensor_resolution: yes
   input_directory: products/tdf/whole_pass/gvar
   input_files: 20*.gvar
   remap_variables: gvar_ch*
   output_template: %yyyy.%mmdd.%hhmm.%satel.gvar
   save_directory: products/tdf/Local/gvar/level1
   save_files: 20??.????.????.*.gvar
}
```

Fig. 1. Gvar.local

B. *ImageMagick*

ImageMagick is software that lets you create, modify, and/or convert images to different formats. This software allows you to resize, distort, and perform other transformations to an image. ImageMagick is controlled command line inputs which may be implemented from different programming languages. There are also GUIs such as MagickWand for PHP or Magick++ (C++) which interface with the software. [11][12]

In this project ImageMagick was used to convert the image from TIFF to JPEG format. It was also used to resize, copy, and rename the image files. The directories that it copied the images to were actual, medium, low, and thumbnail.

C. *PHP*

PHP Hypertext Preprocessor (PHP) is an open source that is widely used around the world; its main use is scripting languages. [15]. PHP was used in this project to construct and execute MySQL commands for the database operations and to execute ImageMagick operations via a command line [13]. PHP is processed by the PHP server and generates Hypertext Markup Language (HTML) to be delivered back to the client [15].

PHP was the successor of PHP/FI. In 1994 Rasmus Lerdorf created PHP using Common Gateway Interface which was written in the C programming language. At first he used PHP for his personal usage like with “Personal Home Page Tools”. When it started becoming larger and Lerdorf saw that PHP can do database interaction, Lerdorf released it to the public so anyone could use it [16].

PHP’s main use is to make web pages interactive and enter information into databases and has other uses as well. It can be used as a calculator to determine what day it is or it can also be used in the execution of math equations. PHP is also used to collect information from users and create graphics using the Graphics Draw (GD) Library.

PHP was used in this project to construct and execute MySQL commands for the database operations and to execute ImageMagick operations via a command line. PHP is a cross platform language making it useful for websites that may change server operating systems.

D. *MySQL*

My Structured Query language (MySQL) is the most popular Open Source SQL database management system. It is developed, distributed, and supported by Oracle Corporation [18].

It is database is a structured collection of data; it may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as part of other applications [17].

E. *Dreamweaver*

Adobe Dreamweaver is a software package that has to be purchased; it is used to create web pages, interactive web pages and to interact with databases. Dreamweaver allows you to write in scripting languages such as PHP and JavaServer Pages. This software also uses server-side scripting languages such as JavaScript [19]. The company that first released Adobe Dreamweaver was Allaire Systems, a computer company in Minnesota, in December of 1997. In 2001 the company Macromedia bought out Allaire, and Macromedia was bought out by Adobe in 2005. Before Adobe bought Dreamweaver it went through revisions [20].

In this project Adobe Dreamweaver was used to build the TeraScan database using the PHP language and to document this project.

F. *phpMyAdmin*

phpMyAdmin is a free software tool written in PHP, intended to handle the administration of MySQL over the web. phpMyAdmin supports a wide range of operations on MySQL, MariaDB and Drizzle. Frequently used operation (managing database, tables, columns, relations, indexes, users, permissions. etc) can perform the user interface, while retaining the ability to directly execute any SQL statements [23][24].

phpMyAdmin was utilized in this project to build and maintain the TeraScan database. Fields and tables within the previously established database were created and modified utilizing this online tool. Actual data can also be modified, created, and deleted from within phpMyAdmin.

G. *iCal*

The calendar application in Mac OS X is known as iCal in Mac OS X and earlier operating system releases. iCal provides tools to keep track of schedules, engagements, reminders, and other important events [25]. This project utilized iCal to trigger the scripts used to process the TeraScan images on a daily basis.

III. GOALS

This research project had three goals for this academic year tasking:

1. Determine and repair the cause of the script failures on the CERSER server.
2. Develop TeraScan script that will process the satellite data into an image, provide overlay information, and place the completed image onto the server.
3. Rewrite the PHP processing script on the CERSER server to parse Geostationary Operational Environmental Satellite (GOES) information into a database and to execute commands to convert and resize the same image.

IV. METHODOLOGY

A. *TeraScan Modification Script*

TeraScan uses five Geostationary Operational Environmental Satellites (GOES) that are around the earth. These satellites include GOES-12 (GOES-East), GOES-10 (GOES-West), GMS-5, Meteosat-7, and Meteosat-5 [5]. GOES Satellites orbit the Earth at the equator and they travel at the speed of that matches Earth’s rotation during their orbit. Also, GOES sends images on a fixed schedule, which includes whole passes [8]. The GOES has an imager, which can display six distinct channels or bandwidths to the user via the transmitted data from the satellite. The SST is a product that is a product of the channels.

- Channel 1 (visible) - Cloud cover and surface features during the day
- Channel 2 (Infrared) - Low cloud/fog and fire detection
- Channel 3 (Infrared) - Upper-level water Vapor
- Channel 4 (Thermal Infrared) - Surface or cloud top temperature
- Channel 6 (Thermal Infrared) - Carbon dioxide band: Cloud detection [5]

In the past, CERSER has focused on receiving data information from the POES rather than GOES. Recently, the hardware was installed to receive data information from both the GOES and POES. The GOES system is now receiving two GOES satellites plus one POES satellite. CERSER is able to receive the data information from GOES using the TeraScan TAC-92 receiver, which stores and processes incoming data to a Linux-based system.

In order to initiate the process, the terminal window in the users v PC or Mac must first be opened. After opening the terminal window, the user has to enter into the TeraScan server. The user login information includes ssh (space) – Y(space) first initial last name@IP address to login into the TeraScan Server (first initial and last name should be together when typed). The server will the request that the user provide a password. After typing the correct password into TeraScan, the user should the type “launchpad” in their login and press enter. After typing “launchpad”, the menu to TeraScans GUIs should open. The user should select TeraMaster and create an area of interest (AOI) which includes increasing/decreasing the area size, pixel size, and density of the AOI. After creating a new master in TeraMaster, the user should go to “File” and “Save As” and then search for the newly created file and save it in their respective directory. Following that, the user should close TeraMaster and return to the terminal window.

In the terminal window, go to “configproc” and in the “configproc” directory change the parameters to match the AOI that was created using TeraMaster. Save the changes in the “configproc” directory. Then go to “launchpad” and click on TeraVision. In TeraVision go to “File” and “Open Data Shelf”. In “Open Data Shelf” go to “Edit Data Library” and go to “New” under Shelves. In “New” find the “configproc” directory in which the images will be saved when a pass is made. Name your new Data Shelf and save it. After saving, go back to “File” and “Open Data Shelf” and click on the new Data Shelf and select images according to what time is desired. After selecting the respectable images, choose the desired channels and once the image(s) are on the TeraVision window, you can edit the images by adding Layers, borders, Grid Overlays, etc. Then click on the Save Icon and Save the image as desired. Then exit TeraVision and exit the terminal window.

B. *CERSER Server Script Failures*

Since there was a new satellite programed this year, the script for the database that was redesigned back in 2011 [10] needed to be updated in order to process the new satellite information being received. It was found that the permissions of the directories on the CERSER server were modified to READ ONLY when the files were migrated to the new server. The directory permissions that were on the CERSER server needed to be changed from READ ONLY to READ & WRITE so the images can be processed.

A remote computer located in Lane Hall gave access to the CERSER server in the E. V. Wilkins Computing Center. To change the directory permission, first right click on the directory and select GET INFO. Scroll to the SHARING & PERMISSIONS section and change each privilege from READ ONLY to READ & WRITE. This procedure was performed on all the directories in the “archives” directory. This action allowed the script to modify and copy images to these directories.

C.*Serverside Script*

Once it was determined that a PHP script was functional on the CERSER server the processing script could now be modified. The languages used in the script are PHP, MySQL, and HTML. The script was first developed in 2006 as an Active Server Page (ASP) on a Windows based server. When the server pages were migrated to a Macintosh server in 2010, the script was rewritten using PHP and MySQL due to the incompatibility of ASP with Macintosh [10].

In 2013 TeraScan capabilities were upgraded. The ground station upgrades included a 3.7m X/L band, a 3.6m C band, and a 5.0m L band dishes, along with accompanying computing hardware [2]. The goal of this project was to update the CERSER script to process images received from the GOES imager by the TeraScan system. This script involved parsing the title into the CERSER database and then copying, converting, renaming, and resizing the original image.

D. *Parse Title to Database*

The title of the image holds the information needed for the database. Image date, time, type of satellite, the channel and the file extension are all found in the title. The GOES image titles have 36 characters including periods and underscores (example: 2014.0302.2001.goes-13.gvar_ch3.tiff). The position of these characters is critical to parsing the elements into the correct database fields. The title is broken down by the following positions:

Characters	**Data**
1-4	Year
6-7	Month
8-9	Day
11-14	Time (Z)
16-22	Satellite Name
24-31	Product (band)
32-36	File Extension (.tiff)

The PHP script parses the title into variables utilizing the “substr” command. This command has the form “substr (string $string , int $start [, int $length])” where $string is the input string, $start is the integer for the starting character in the string, and $length is the integer for the total number of characters. It must be remembered that the starting integer starts counting at “0” for the first character in the string.

In the script shown in Appendix A, Figure 2, the title of the file has been previously placed into the PHP variable “$satName.” The script first tests if the image is from the GOES imager by comparing characters 15-18 [substr($satName, 15, 4)] to the string “goes”. If they are equal, then the script proceeds to parse the title.

The first image element to be entered into the database is the satellite name found in line 70 of Appendix A, Figure 3. This is extracted from the title and placed into the variable “sat”.

The process then converts the date into the format MM/DD/YYYY utilizing the PHP concatenation “.” (period) as shown in Appendix A, Figure 4. This date is then stored in “cnvrtDate”.

The time is then placed into the variable $cnvrtTime with the character “z” added to denote the time zone (Appendix A, Figure 5).

The final parsing element shown in Appendix A, Figure 6, is the $product variable. The script checks the character located at position 30 to identify which band of the imager was used to develop the satellite image. The options along with the text string to be inserted into the database are as follows:

Option	Text String
"1"	Channel 1 Visible 0.52-0.72 um
"2"	Channel 2 Infrared 3.78-4.03 um
"3"	Channel 3 Vapor 6.47-7.02 um
"4"	Channel 4 Upper Vapor 10.2-11.2 um
"6"	Channel 6 Thermal IR 12.9-13.7 um
"S"	Sea Surface Temperature

Once the file title is parsed into the assorted variables, the variables are combined into an SQL statement (line 111, Appendix A, Figure 7) and placed into the variable "$query". The variable is then written to the screen using the "echo" command for viewing by the developer. This step can be removed, but was left in as a feedback mechanism for the script.

The next step is to execute the SQL statement using the PHP command mysql_query() (Line 117, Appendix A, Figure 7). This will place the variables into the appropriate fields of a new database record in the "images" table. The results of this query are placed into the variable "$result" so that any failures can be displayed to the user.

E. *Resize/Rename/Copy IMG*

The next group of operations involves resizing, renaming, copying, and converting the TeraScan images from TIFF to JPEG formats. PHP is used to execute command line instructions to the graphic program ImageMagick. This section of code was not modified during this project, but a brief overview of the operation follows.

Appendix A, Figure 8 shows the code that transfers the record id to the variable $lastID. The function "mysql_insert_id()" is the PHP code that retrieves the record id. The $lastID will be the new name for the record. When a record is inserted into the database the key id is automatically created and incremented. There are no records that have the same key id.

Appendix A, Figure 9 is the section of code required to manipulate the various images produced by the TeraScan system. These processes use a command line execution of modules found in the open source graphic program ImageMagick. This program has many capabilities such as resizing, flipping, rotating, distorting, and transforming images.

The first step in this process is assigning the current name of the TeraScan image and its location to the variable $orgFile. Using the $lastID variable is used to assign the new title for the converted files in line 138 of Appendix A, Figure 9 to the variable $newFile. These variables are written to the screen for confirmation in line 141.

The first is to copy the TeraScan image from the "original" directory to the "actual" directory with its new title derived from the variable $lastID (lines 148-149). The parameters for each of these steps is placed into the variable $cnvrt and then used in the PHP function exec() which is used to execute command line instructions.

These steps are repeated for the medium, low, and thumbnail conversions using various parameters to convert, resize, and rename the images. Once this conversion is complete, the PHP command unlink() is utilized to delete the original file. The script then begins processing the next image in the "original" directory until all have been processed.

V. RESULTS

A. *TeraScan*

Images are able to be processed using both TeraVision and the UNIX script. In TeraVision, the user can edit to have borders, layers, Grid Overlays, etc. In the script, each time a pass comes in and is converted into data, the batch.ingest takes the pass and cleaves it to match the parameters of the picture in the gvar.local directory, which includes the cover_area, input files, the save directory, and other parameters.

B. *Server/ Script*

After changing the permissions of the picture archives directories on the CERSER server in E. V. Wilkins Computing Center, the images can now be modify to be used in the database. The images are now able to process and to be displayed on the database. The information from parsing the title of the images is now being put into the database. The code in the script for resizing/ renaming/ copying the image is still working and larger images are able to be viewed when selected from the web page viewer.

VI. CONCLUSIONS

Images can be successfully processed however; there are other goals that were not met. After changing the parameters in the script to modify the images to match the AOI, there needs to be a way to transfer the images to the CERSER server in order that CERSER can get the images automatically from the TeraScan server. Current lt images must be processed manually and sent to the CERSER server. There needs to be a daily automated process written the script to receive processed and finalized images from the TeraScan server to send to the CERSER server. Also, there needs to be an increase of image resolution with the images being received and processed through the script in TeraScan.

Overall the database is a success and there have not been any more problems with the images transition from the satellite to E. V. Wilkins Computing Center to the database.

VII. FUTURE WORK

A daily automated process of images from the TeraScan server to the CERSER server should be developed next. Increased resolution of the images should be explored to make the image useful for research.

The PHP script on the CERSER server will have to be updated once the POES hardware comes back on line. This should be a minor revision if the TeraScan script remains the same. Once more channels or satellites are added to the TeraScan output, the script will need to be expanded.

The TeraScan viewer capabilities on the CERSER server should be expanded to allow for searching and sorting of the information. A target audience should be identified and research should be completed to see how other ground stations are presenting and utilizing their data.

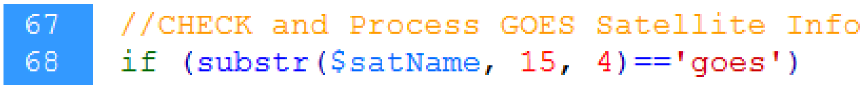
ACKNOWLEDGEMENTS

We will like to thank Dr. Linda Hayden for supporting our 2014 research. We like to thank Andrew Brumfield and Seaspace for helping with doing the research.

REFERENCES

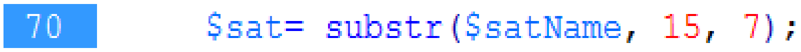
1. "TeraScan 4.2" Internet: ecsu-goes-east/terascan/thelp/home_basic/what_is_terscan.html, n.d. [Apr. 2, 2014]
2. "CERSER Facilities" Internet: http://cerser.ecsu.edu/facilities.html, n.d. [Apr. 2, 2014]
3. "TeraScan 3.2" Internet: http://marine.rutgers.edu/~lojanen/terascan/html/home_basic/terascan_guis.html#TeraVision, Sept. 10, 2002 [Apr. 2, 2014]
4. "Geostationary Satellites" Internet: http://marine.rutgers.edu/~lojanen/terascan/html/home_basic/geo_sats_overview.htm, n.d. [Mar. 4, 2014]
5. "GOES Imager Instrument" Internet: http://noaasis.noaa.gov/NOAASIS/ml/imager.html, n.d. [Apr. 2, 2014]
6. "TeraScan 3.2" Local Drive, http://ecsu-goes-east/terascan/thelp/batch_processing/GVAR_configprocs.html , May 10, 2002 [Mar. 4, 2014]
7. "TeraScan 3.2" Local Drive, http://ecsu-goes-east/terascan/thelp/man7/gvar.html, May 10, 2002 [Mar. 4, 2014]
8. "What Are Geostationary Satellites?" Internet: http://marine.rutgers.edu/~lojanen/terascan/html/home_basic/geo-info.htm. Jun. 6, 2006 [Mar 12, 2014]
9. "GVAR System Overview" Internet: http://www.ssec.wisc.edu/mcidas/doc/sdi_man/jul00/SDIHTML-123.HTML, July 2000 [Mar. 4, 2014]
10. Luke, Autumn. Bly, Patrina. "Redesign of CERSER Cataloguing System for TeraScan Processed Images." Internet: http://nia.ecsu.edu/ur/1011/teams/mmt/mmt1011-paper.pdf , n.d. [Mar. 12, 2014].
11. Cristy. "ImageMagick History."Internet: http://www.imagemagick.org/www/history.html, n.d.[Feb. 11, 2014]
12. Tyssen, Anthony. "Examples of ImageMagick Usage(Version 6)." Internet: http://www.imagemagick.org/Usage/, Nov. 30, 2012 [Feb. 11, 2014]
13. "Convert, Edit, and Compose Images." Internet: http://www.imagemagick.org/, n.d. [Feb. 11,2014]
14. John, Michael S. "ImageMagick-some cool things you can do with it!" Internet: http://ubuntuvideocast.com/2012/05/25/imagemagick-some-cool-things-you-can-do-with-it/, May 25, 2012 [Feb. 11, 2014]
15. "What is PHP?" Internet: http://php.net/manual/en/intro-whatis.php, n.d. [Feb. 14, 2014]
16. Grubb, Kristen. "What Does PHP Stand For?" Internet: http://www.brighthub.com/internet/web-development/articles/62713.aspx, Aug. 3, 2012 [Feb. 14, 2014]
17. "History of PHP" Internet: http://php.net/manual/en/history.php.php, n.d. [Feb. 14, 2014]
18. Bradley, Angela "What is PHP Used For?" Internet: http://php.about.com/od/phpbasics/qt/what_is_php_used_for.htm, n.d. [Feb. 14, 2014]
19. "Help saving MySQL" Internet: http://monty-says.blogspot.com/2009/12/help-saving-mysql.html, Dec. 12,2009 [Mar. 25, 2014]
20. "1.3.1. What is MySQL?" Internet: http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql.html, n.d. [Mar. 25, 2014]
21. "Dreamweaver system requirements" Internet: http://web.archive.org/web/20090517002426/http:/kb2.adobe.com/cps/191/tn_19124.html, n.d. [Mar. 25, 2014]
22. "Adobe Completes Acquisition of Macromedia" Internet: www.adobe.com/aboutadobe/pressroom/pressreleases/pdfs/200512/120505AdobeAcquiresMacromedia.pdf, Dec. 5, 2005 [Mar. 25, 2014]
23. "Learn to build dynamic websites and web applications" Internet: www.adobe.com/devnet/dreamweaver/application_development.html, n.d. [Mar. 25, 2014]
24. "Dreamweaver/ About Dreamweaver sites" Internet: help.adobe.com/en_US/dreamweaver/cs/using/WS753df6af718a350a5601124413168b2b12f-8000.html, n.d. [Mar. 25, 2014]
25. "phpMyAdmin-About" Internet: www.phpmyadmin.net/home_page/about.php, n.d. [Mar. 25, 2014]
26. "PhpMyAdmin Tutorial" Internet: www.siteground.com/tutorials/phpmyadmin/, n.d. [Mar. 25, 2014]

APPENDIX A



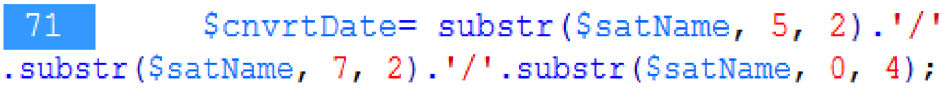
```
//CHECK and Process GOES Satellite Info
if (substr($satName, 15, 4)=='goes')
```

Fig. 2. Checking for a GOES image



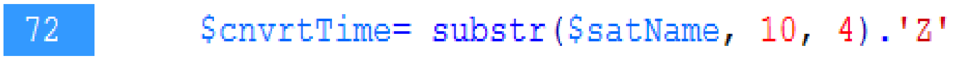
```
        $sat= substr($satName, 15, 7);
```

Fig. 3. Extracting the satellite name



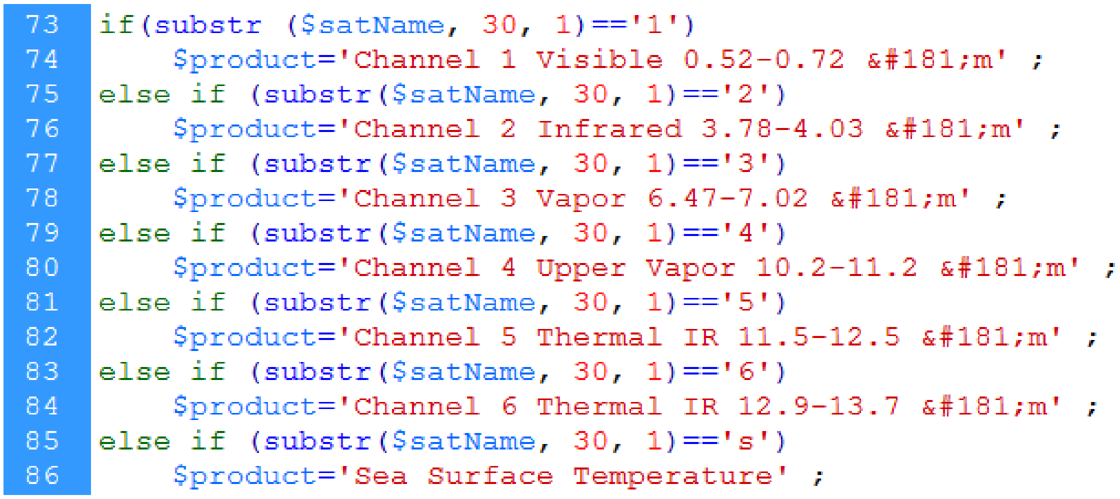
```
        $cnvrtDate= substr($satName, 5, 2).'/'
.substr($satName, 7, 2).'/'.substr($satName, 0, 4);
```

Fig. 4. Converting and storing the image date



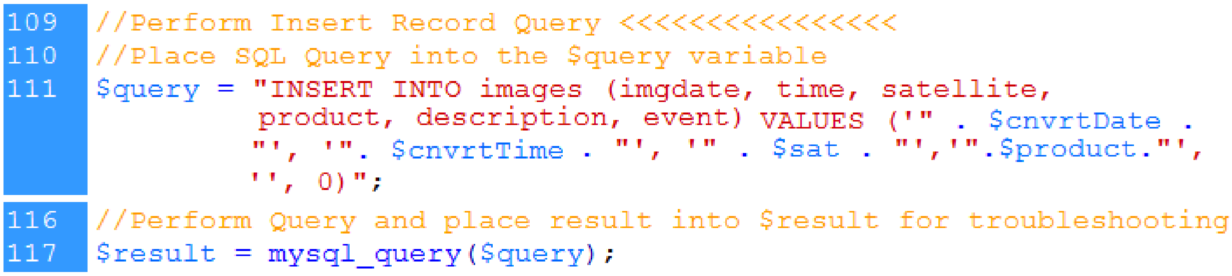
```
        $cnvrtTime= substr($satName, 10, 4).'Z'
```

Fig. 5. Extracting the GMT time



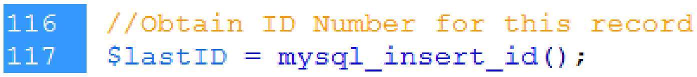
```
if(substr ($satName, 30, 1)=='1')
    $product='Channel 1 Visible 0.52-0.72 &#181;m' ;
else if (substr($satName, 30, 1)=='2')
    $product='Channel 2 Infrared 3.78-4.03 &#181;m' ;
else if (substr($satName, 30, 1)=='3')
    $product='Channel 3 Vapor 6.47-7.02 &#181;m' ;
else if (substr($satName, 30, 1)=='4')
    $product='Channel 4 Upper Vapor 10.2-11.2 &#181;m' ;
else if (substr($satName, 30, 1)=='5')
    $product='Channel 5 Thermal IR 11.5-12.5 &#181;m' ;
else if (substr($satName, 30, 1)=='6')
    $product='Channel 6 Thermal IR 12.9-13.7 &#181;m' ;
else if (substr($satName, 30, 1)=='s')
    $product='Sea Surface Temperature' ;
```

Fig. 6. Image product selection



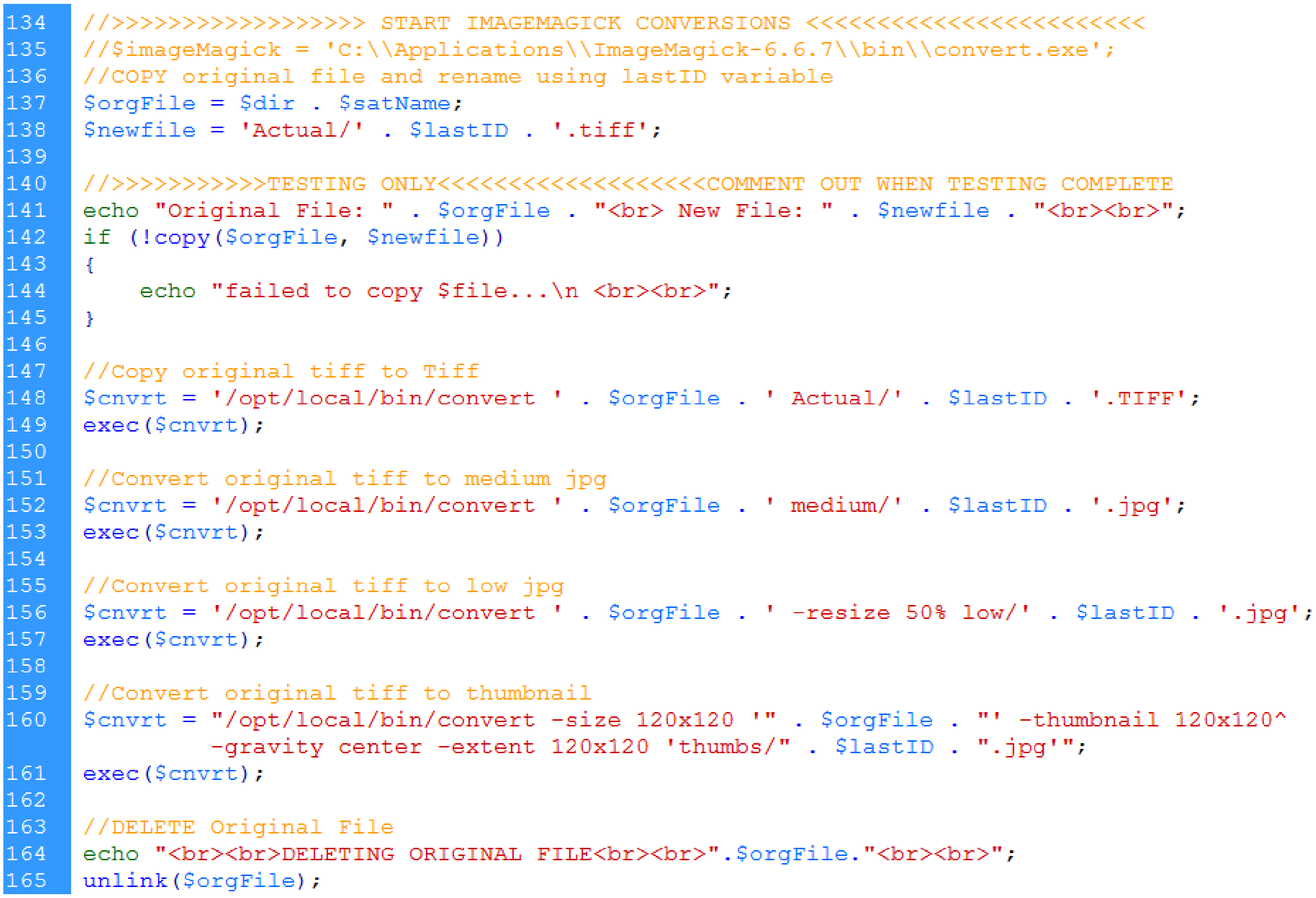
```
//Perform Insert Record Query <<<<<<<<<<<<<<<
//Place SQL Query into the $query variable
$query = "INSERT INTO images (imgdate, time, satellite,
         product, description, event) VALUES ('" . $cnvrtDate .
        "', '". $cnvrtTime . "', '" . $sat . "','".$product."',
        '', 0)";
//Perform Query and place result into $result for troubleshooting
$result = mysql_query($query);
```

Fig. 7. SQL statement and its execution



```
 //Obtain ID Number for this record
 $lastID = mysql_insert_id();
```

Fig. 8. Obtaining the ID number of the record just inserted



```
//>>>>>>>>>>>>>>>>>> START IMAGEMAGICK CONVERSIONS <<<<<<<<<<<<<<<<<<<<<<<<
//$imageMagick = 'C:\\Applications\\ImageMagick-6.6.7\\bin\\convert.exe';
//COPY original file and rename using lastID variable
$orgFile = $dir . $satName;
$newfile = 'Actual/' . $lastID . '.tiff';

//>>>>>>>>>>>TESTING ONLY<<<<<<<<<<<<<<<<<<<COMMENT OUT WHEN TESTING COMPLETE
echo "Original File: " . $orgFile . "<br> New File: " . $newfile . "<br><br>";
if (!copy($orgFile, $newfile))
{
    echo "failed to copy $file...\n <br><br>";
}

//Copy original tiff to Tiff
$cnvrt = '/opt/local/bin/convert ' . $orgFile . ' Actual/' . $lastID . '.TIFF';
exec($cnvrt);

//Convert original tiff to medium jpg
$cnvrt = '/opt/local/bin/convert ' . $orgFile . ' medium/' . $lastID . '.jpg';
exec($cnvrt);

//Convert original tiff to low jpg
$cnvrt = '/opt/local/bin/convert ' . $orgFile . ' -resize 50% low/' . $lastID . '.jpg';
exec($cnvrt);

//Convert original tiff to thumbnail
$cnvrt = "/opt/local/bin/convert -size 120x120 '" . $orgFile . "' -thumbnail 120x120^
         -gravity center -extent 120x120 'thumbs/" . $lastID . ".jpg'";
exec($cnvrt);

//DELETE Original File
echo "<br><br>DELETING ORIGINAL FILE<br><br>".$orgFile."<br><br>";
unlink($orgFile);
```

Fig. 9. Command lines and execution for ImageMagick