6th June 2014

Swiss Space Center

CHEOPS : SOC Preprocessing

optional project in computer science

Prepared by:

Julien Graisse

Checked/Approved/Signed by:

Anton Ivanov

Swiss Space Center
EPFL - Lausanne

06.06.2014

RECORD OF REVISIONS

ISS/REV	Date	Modifications	Created/modified by

RECORD OF REVISIONS
GLOSSARY
INTRODUCTION
THE CHEOPS PROJECT
OBJECTIVES OF THIS PROJECT
DESIGN ASSUMPTION
REQUIREMENTS
The Fits converter
The preprocessor
DEVELOPMENT TOOLS
The programming language
FITS format manipulation
The GUI
Development environment
SOFTWARE IMPLEMENTATION
THE FITS CONVERTER
Interface
classes
THE PREPROCESSOR
DISCUSSION
PROGRESSION
The FITS converter
The preprocessor
CONCLUSION
APPENDIX: DOCUMENTATION

GLOSSARY

ESA: European Space Agency

HK: HouseKeeping, standard exit routine.

GUI: Graphical User Interface. An interface allowing users to interact with a program via visual components in opposition to a command-line interface.

MOC: Mission Operation Center

MVC design pattern: Stands for Model-View-Controller. A pattern consisting of dividing the structure of the code between the *model* (data), the *view* (the display) and the *controller* (the operations).

RAM: Random Access Memory. Memory used by running processes on all common computers

SOC: Science Operarion Center

SSO: Swiss Space Office

UML: Unified Modeling language, a specification language used here to represent the organization of the code.

URL: Uniform Resource Locator, the string address of a remote resource on the web.

INTRODUCTION

THE CHEOPS PROJECT

CHEOPS (Characterising ExOPlanets Satellite) is a mission organized by the SSO and the ESA. It consists of the development of a space telescope dedicated to examining and characterising transiting exoplanets. The telescope, mounted on a s-class satellite platform will be the first mission dedicated to search for transits by means of ultrahigh precision photometry on bright stars already known to host planets.

The project was selected in October 2012 and is planned to be launched in 2017

MISSION SUMMARY

Primary Goal	Characterize transiting exoplanets on known bright and nearby host stars
Targets	Known exoplanet host stars with a V-magnitude < 12.5 (goal: 13) anywhere on the sky
Wavelength	Visible range : 400 to 1100 nm
Telescope	33 cm reflective an-axis telescope
Orbit	Sun-synchronous Low Earth Orbit, LTAN 6am, altitude 620-800 km
Lifetime	3.5 years
Type S-class mission	S-class mission

Source: http://cheops.unibe.ch/index.php/article-2

OBJECTIVES OF THIS PROJECT

The idea of this project is to start the development of a preprocessing subsystem between MOC and SOC. This data interface's main function shall be to format incoming data from the MOC telemetry data storage (namely science, housekeeping and auxiliary data) for utilization at SOC.

This project consists of two tasks. The first one is a FITS converter. Its objective is to fetch and download raw data from a source and to convert it in the standard FITS format used in astronomy. The second task is to develop a preprocessor merging incoming HK data with associated FITS images.

DESIGN ASSUMPTION

REQUIREMENTS

The Fits converter

The different requirements of the first task are listed below.

Name: URL fetching
Type: Functional requirement
Status: draft
Verification method: Test
Parent: None
Description: All FITS files reachable from the provided URL must be downloaded.

Name: FITS conversion
Type: Functional requirement
Status: Issued
Verification method: Test
Parent: None
Description: the output of the conversion must be in the Standard FITS format

Name: Graphical interface implementation
Type: graphical requirement.
Status: Issued
Verification method: Test
Parent: None
Description: All functions of the software must be accessible through a GUI.

Name: MVC integration
Type: architectural requirement
Status: draft
Verification method: Inspection

Parent: Graphical interface implementation

Description: The GUI must be developed respecting the MVC architectural design pattern.

Name: Caching of downloaded data

Type: Performance requirement

Status: Issued

Verification method: Inspection, test

Parent: URL fetching

Description: Hard drive disks being the slowest memory, downloaded raw data must remain on the RAM until completion of the conversion.

Name: Exception handler

Type: Behavioural and maintenance requirements

Status: Issued

Verification method: Inspection

Parent: None

Description: All entries of the user must be checked for correctness and invalid operations must be handled

The preprocessor

The different requirements of the first task are listed below.

Name: Merging

Type: Functional requirement

Status: Issued

Verification method: Test

Parent: None

Description: When a folder is provided, all HK data it contains must be integrated to corresponding FITS files in it.

Name: Writing overload

Type: Performance requirement

Status: Issued

Verification method: Inspection, test

Parent: Merging

Description: The original FITS files must be overwritten by the ones resulting of the merging.

DEVELOPMENT TOOLS

The programming language

It has been decided to work with C++ for the CHEOPS mission.

FITS format manipulation

There already exist libraries to work with the FITS format in C++. Not using them would be a huge waste of time. The one used in this project is ccfits, an object-oriented wrapper of a multi-platform C library, cfitsio.

The GUI

There exist many libraries available in C++. Our criteria for the selection are:

- Multi-platform
- Comprehensiveness
- Open-Source

Library	Multi-platform	Comprehensiveness	Open-Source	Comments
.NET	★☆	★★★	★☆☆	Microsoft's platform. Mainly used under windows
GTK+	★★☆	★★☆	★★★	One of the world's most used graphical libraries
Qt	★★☆	★★★	★★☆	Excellent documentation. Extremely complete libraries: More than a GUI, it's a framework.
wxWidgets	★★☆	★★☆	★★★	Quite complete. Reputed hard to take in hand.
FLTK	★★★	★☆☆	★★★	Designed to be light and portable. Ugly.

For its features, its power and its support, **Qt** is the most suitable framework.

Development environment

OS

The project is developed on linux but as seen previously, it is meant to be portable. No special requirement is needed for the machine. The application has been tested so far on a common laptop with the following characteristics:

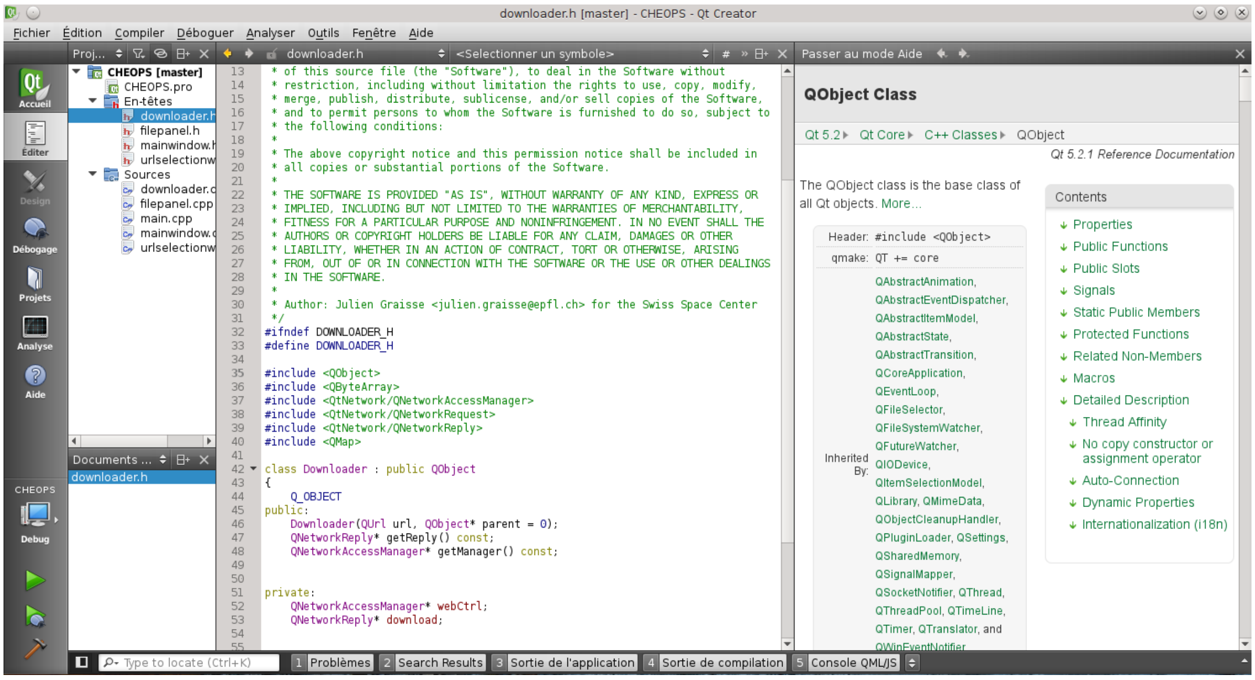
Ubuntu Linux 12.04

intel(r) core(tm) i3-2310m cpu @ 2.10ghz dual-core

4Gb RAM

IDE

The IDE used for this project is Qt Creator, an enhanced environment for C++ coding under Qt.



```
 * of this source file (the "Software"), to deal in the Software without
 * restriction, including without limitation the rights to use, copy, modify,
 * merge, publish, distribute, sublicense, and/or sell copies of the Software,
 * and to permit persons to whom the Software is furnished to do so, subject to
 * the following conditions:
 *
 * The above copyright notice and this permission notice shall be included in
 * all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
 * FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS
 * IN THE SOFTWARE.
 *
 * Author: Julien Graisse <julien.graisse@epfl.ch> for the Swiss Space Center
 */
#ifndef DOWNLOADER_H
#define DOWNLOADER_H

#include <QObject>
#include <QByteArray>
#include <QtNetwork/QNetworkAccessManager>
#include <QtNetwork/QNetworkRequest>
#include <QtNetwork/QNetworkReply>
#include <QMap>

class Downloader : public QObject
{
    Q_OBJECT
public:
    Downloader(QUrl url, QObject* parent = 0);
    QNetworkReply* getReply() const;
    QNetworkAccessManager* getManager() const;


private:
    QNetworkAccessManager* webCtrl;
    QNetworkReply* download;

```

QObject Class

Qt 5.2 ▸ Qt Core ▸ C++ Classes ▸ QObject

Qt 5.2.1 Reference Documentation

The QObject class is the base class of all Qt objects. More...

Header:	#include <QObject>
qmake:	QT += core
Inherited By:	QAbstractAnimation, QAbstractEventDispatcher, QAbstractItemModel, QAbstractState, QAbstractTransition, QCoreApplication, QEventLoop, QFileSelector, QFileSystemWatcher, QFutureWatcher, QIODevice, QItemSelectionModel, QLibrary, QMimeData, QObjectCleanupHandler, QPluginLoader, QSettings, QSharedMemory, QSignalMapper, QSocketNotifier, QThread, QThreadPool, QTimeLine, QTimer, QTranslator, and QWinEventNotifier

Contents

- Properties
- Public Functions
- Public Slots
- Signals
- Static Public Members
- Protected Functions
- Related Non-Members
- Macros
- Detailed Description
  - Thread Affinity
  - No copy constructor or assignment operator
  - Auto-Connection
  - Dynamic Properties
  - Internationalization (i18n)

Fig: Qt Creator

FITS reader

The software selected to read and test FITS files is fv, an easy to use and multi-platform FITS viewer and editor developed by the NASA.

Code redistribution

This project is under the GNU GPLv3 licence. The source code is accessible on github at the following address:

https://github.com/merodrem/CHEOPS

SOFTWARE IMPLEMENTATION

THE FITS CONVERTER

Interface

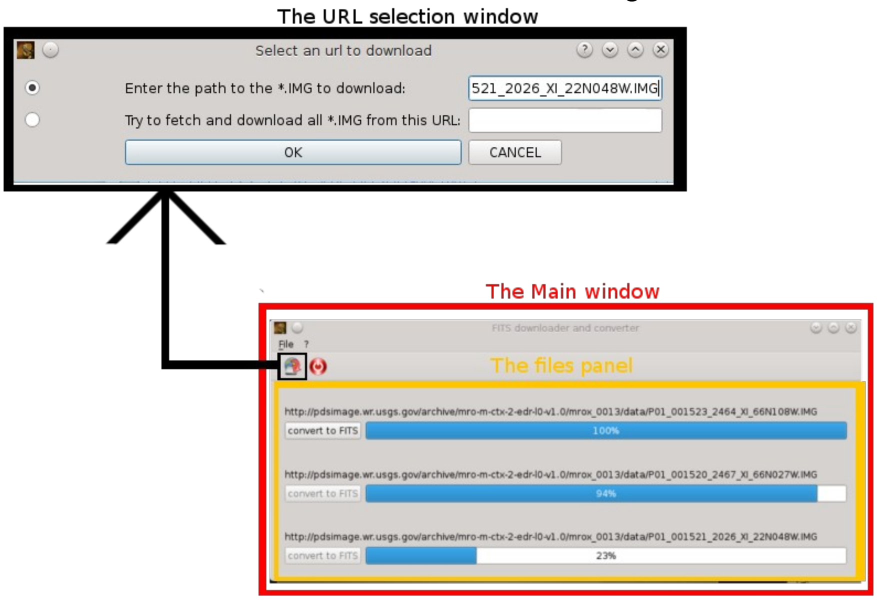
The interface can be divided in 3 modules:

The main window: Containing and coordinating all sub-modules.

The URL selection window: Opened from the main window. It offers the user to enter either:

- An URL leading to a .fits file
- An URL leading to a page containing link to .fits files. In that case the software cares of downloading all files.

The files panel: Contained in the main window. It handles each file being downloaded



Enter the path to the *.IMG to download: 521_2026_XI_22N048W.IMG

Try to fetch and download all *.IMG from this URL:

http://pdsimage.wr.usgs.gov/archive/mro-m-ctx-2-edr-l0-v1.0/mrox_0013/data/P01_001523_2464_XI_66N108W.IMG

http://pdsimage.wr.usgs.gov/archive/mro-m-ctx-2-edr-l0-v1.0/mrox_0013/data/P01_001520_2467_XI_66N027W.IMG

http://pdsimage.wr.usgs.gov/archive/mro-m-ctx-2-edr-l0-v1.0/mrox_0013/data/P01_001521_2026_XI_22N048W.IMG

classes

The architecture respects the MVC philosophy. However, The model (really simple here) and view have been merged for clarity reasons.

<u>Controller classes:</u>

downloader.h: Downloads and converts files.

<u>Model/view classes:</u>

mainwindow.h

filepanel.h

urlselectionwindow.h

The dependencies between classes can be visualized by an UML diagram:

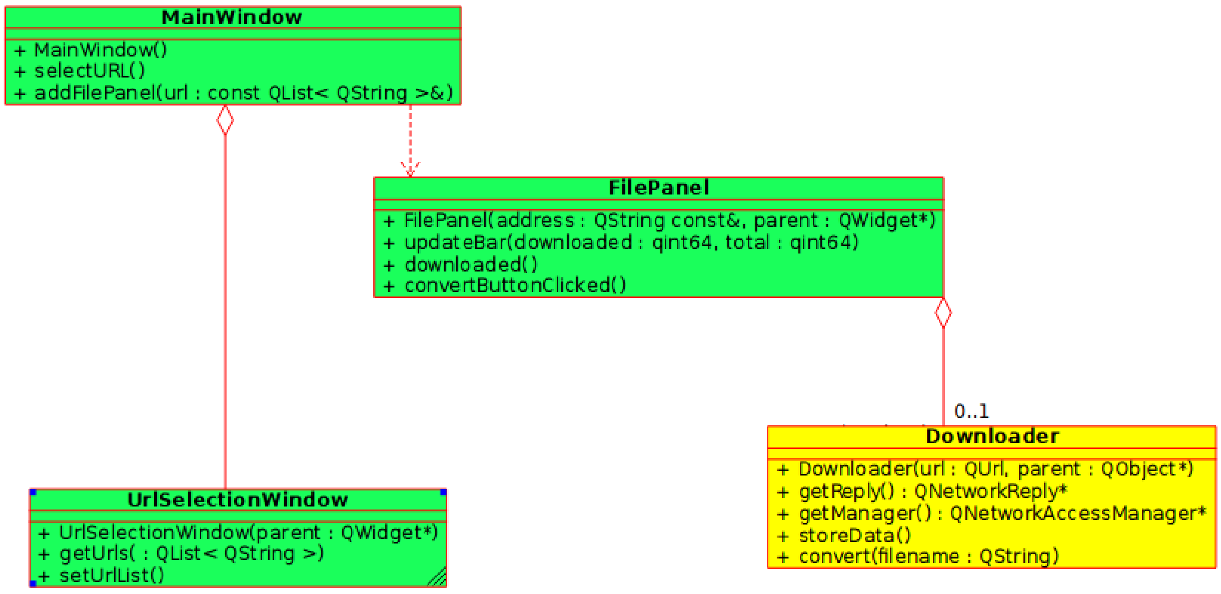

MainWindow
+ MainWindow()
+ selectURL()
+ addFilePanel(url : const QList< QString >&)
FilePanel
+ FilePanel(address : QString const&, parent : QWidget*)
+ updateBar(downloaded : qint64, total : qint64)
+ downloaded()
+ convertButtonClicked()
0..1
Downloader
+ Downloader(url : QUrl, parent : QObject*)
+ getReply() : QNetworkReply*
+ getManager() : QNetworkAccessManager*
+ storeData()
+ convert(filename : QString)
UrlSelectionWindow
+ UrlSelectionWindow(parent : QWidget*)
+ getUrls(: QList< QString >)
+ setUrlList()

Fig: UML diagram of view classes(green) and controller classes (yellow)

THE PREPROCESSOR

For now, the controller is a simple command line application rather aimed to be integrated in other Qt softwares than to become a stand-alone application. For that purpose, Qt's libraries are used as much as possible even though no GUI has been implemented.

The program's main feature consists of the function:

void merge(QString directory, QString dataFile);

where **dirtectory** is the path to a directory containing FITS files and **dataFile** the path to the HK data. As this function returns nothing, it is said to have side effects: Namely here, the overwritting of FITS files with data tables.

The current implementation enters manually some constants in the code. In the next steps of development, thoses values should be asked to the user, or defined in a file apart using a protocol.

Those constants concern:

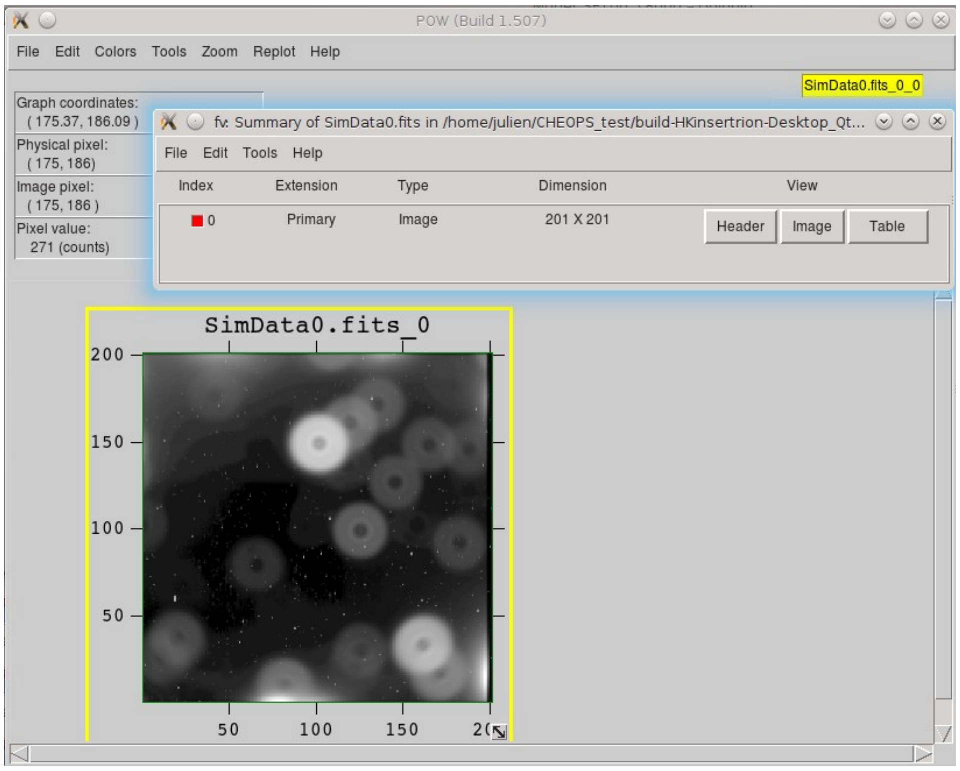
-The number of stars registered. Set to 100

-The number of images taken per FITS file. Set to 60

-The number different data recorded. Set to 19.

Exemple of execution

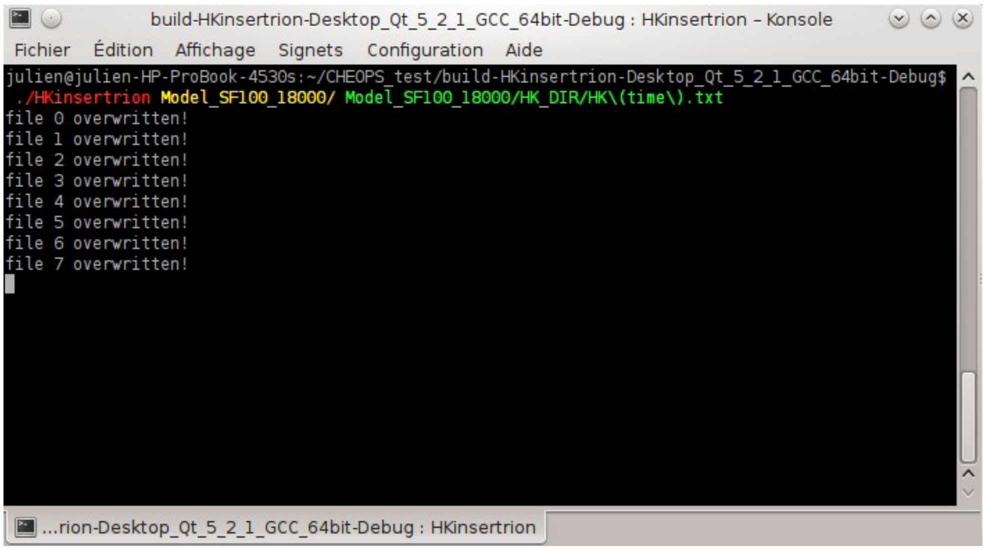
At the initial state the files contained in the folder Model_SF100_18000/ (a folder only contain a picture. This is the file SimData0.fits:



Index	Extension	Type	Dimension	View
🟥 0	Primary	Image	201 X 201	Header Image Table

Fig: An overview of the file SimData0.fits from fv

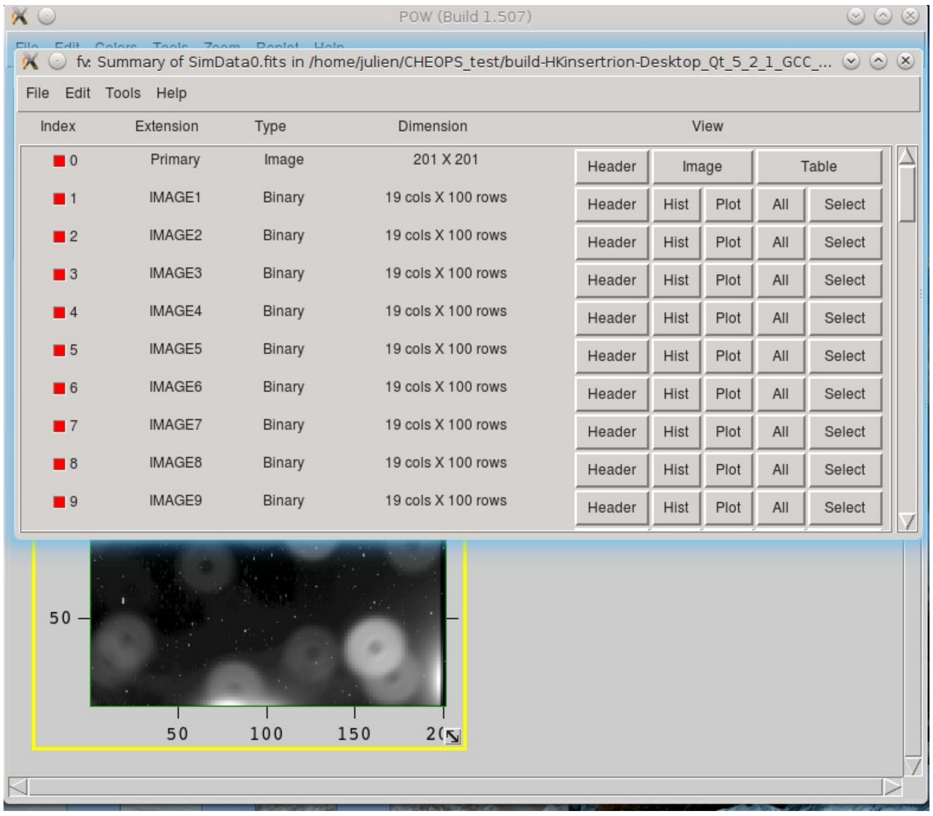
The preprocessor is launched from a command line:


build-HKinsertrion-Desktop_Qt_5_2_1_GCC_64bit-Debug : HKinsertrion – Konsole
Fichier Édition Affichage Signets Configuration Aide
...rion-Desktop_Qt_5_2_1_GCC_64bit-Debug : HKinsertrion

```
julien@julien-HP-ProBook-4530s:~/CHEOPS_test/build-HKinsertrion-Desktop_Qt_5_2_1_GCC_64bit-Debug$
 ./HKinsertrion Model_SF100_18000/ Model_SF100_18000/HK_DIR/HK\(time\).txt
file 0 overwritten!
file 1 overwritten!
file 2 overwritten!
file 3 overwritten!
file 4 overwritten!
file 5 overwritten!
file 6 overwritten!
file 7 overwritten!
```

Fig: The command (in red), the folder (in yellow) and the path to the data file (in green)

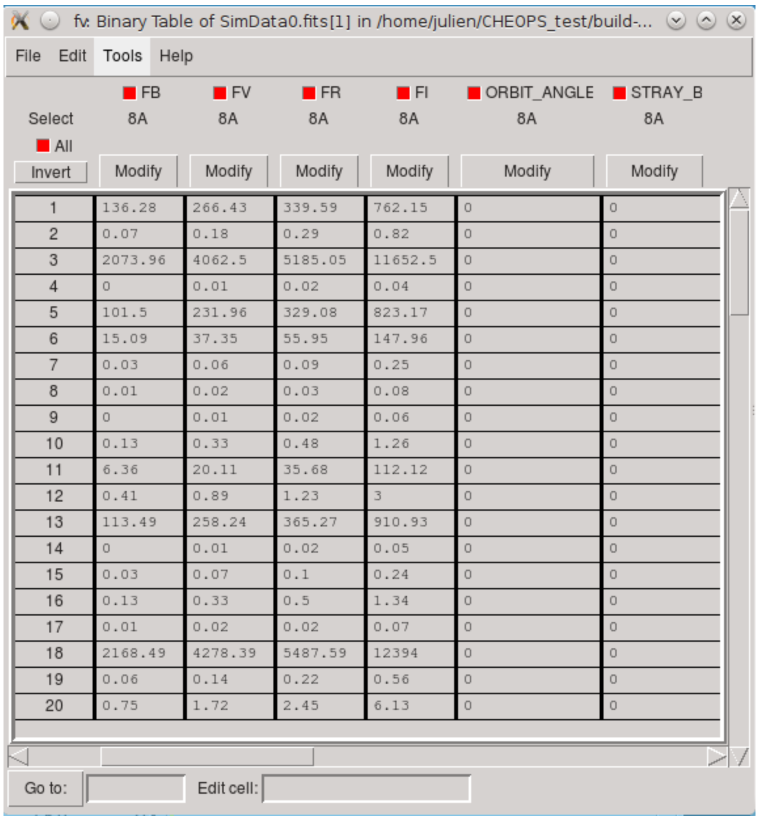
When done, all the files of the folder contain one table per image (60 in total)


POW (Build 1.507)
File Edit Colors Tools Zoom Replot Help
fv: Summary of SimData0.fits in /home/julien/CHEOPS_test/build-HKinsertrion-Desktop_Qt_5_2_1_GCC_...
File Edit Tools Help
50
50 100 150 2(

Index	Extension	Type	Dimension	View
🟥 0	Primary	Image	201 X 201	Header Image Table
🟥 1	IMAGE1	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 2	IMAGE2	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 3	IMAGE3	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 4	IMAGE4	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 5	IMAGE5	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 6	IMAGE6	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 7	IMAGE7	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 8	IMAGE8	Binary	19 cols X 100 rows	Header Hist Plot All Select
🟥 9	IMAGE9	Binary	19 cols X 100 rows	Header Hist Plot All Select

Fig: Back to SimData0.fits after the execution

All those tables having the following form:


fv: Binary Table of SimData0.fits[1] in /home/julien/CHEOPS_test/build-...
File Edit Tools Help
Select
🟥 All
Invert
Modify
Go to:
Edit cell:

Select	🟥 FB 8A	🟥 FV 8A	🟥 FR 8A	🟥 FI 8A	🟥 ORBIT_ANGLE 8A	🟥 STRAY_B 8A
1	136.28	266.43	339.59	762.15	0	0
2	0.07	0.18	0.29	0.82	0	0
3	2073.96	4062.5	5185.05	11652.5	0	0
4	0	0.01	0.02	0.04	0	0
5	101.5	231.96	329.08	823.17	0	0
6	15.09	37.35	55.95	147.96	0	0
7	0.03	0.06	0.09	0.25	0	0
8	0.01	0.02	0.03	0.08	0	0
9	0	0.01	0.02	0.06	0	0
10	0.13	0.33	0.48	1.26	0	0
11	6.36	20.11	35.68	112.12	0	0
12	0.41	0.89	1.23	3	0	0
13	113.49	258.24	365.27	910.93	0	0
14	0	0.01	0.02	0.05	0	0
15	0.03	0.07	0.1	0.24	0	0
16	0.13	0.33	0.5	1.34	0	0
17	0.01	0.02	0.02	0.07	0	0
18	2168.49	4278.39	5487.59	12394	0	0
19	0.06	0.14	0.22	0.56	0	0
20	0.75	1.72	2.45	6.13	0	0

DISCUSSION

The FITS converter

Concerning its basic purpose, the software is finished. But it is still quite rudimentary and as in many applications, new features and improvements could be added. Namely:

-Reorganize the layout of the application: Not all widgets resize correctly when the window is adjusted

-Optimize the refresh of the window: Some widgets are currently deleted then redrawn alike at each update.

-Optimize the download speed: When downloading from this application, the bitrate is statistically around 10% lower than when downloading from a usual browser (tested on mozilla firefox and Google Chrome). Tests should be carried out to determine if the problem comes from Qt's Network module or a bad management of the threads within the application.

-Many more options to add: Saving a downloaded file without converting it, open a binary file on the drive to convert it, adding additional data, to the file,...

-Complete the readme for next programmers

-Create a small documentation for developed classes

-Export the code on MAC OS and Windows

The preprocessor

This project offers only the first stone of the whole preprocessing system. This program is a first try to merge data with FITS files. At this stage, the tests are carried out on simulation data with a lack of perspective. It is more than likely that the code doesn't fit the final needs. But implementing a first algorithm was a milestone.

Bugs and weaknesses of the current code include:

-To be too much hard-coded. Some parameters such as the number of images per file shouldn't be set directly in the code, but rather asked to the user or written in a file apart.

-To rely to much on the structure of the code. If for example the FITS file's names don't follow the model *SimData[X].fits* anymore, the program won't overwrite them even though their folder is provided.

-To be not optimized enough. For example, reading files using regular expressions could spare some tedious work.

-To be too sensible to exceptions.

CONCLUSION

This project was a first step in the development of a SOC preprocessing system for CHEOPS. The idea was to build tools working on the FITS format for next tasks. Those tools are namely a FITS converter used to build files in a standard FITS format from raw data, and an early preprocessor in charge of integrating data to corresponding FITS files.

The tasks have been developed in C++ on ubuntu linux using the framework Qt. Time was spared using an already existing library handling the FITS format: cfitsio. Although special care was taken to only select multi-platform technologies to keep the code portable, no code has been recompiled and tested on other operating systems. As a general comment about the progress, both tasks are functional but must be refined in the next steps.

This semester optional project was a bit to small to cover the whole system and over all a bit too early in the mission's development stage. lots of features, rules, organization,..., were still to be determined. The development was often pending waiting for available resources to test or even went in the wrong direction by using Python before it were decided to use C++. Nevertheless, this work represent a good start for further developments. It was also a good grip and learning of the FITS format and how to handle it from a programmer's point of view.

APPENDIX: DOCUMENTATION

Institutions

The CHEOPS Project

http://cheops.unibe.ch/

The Swiss Space Center

http://space.epfl.ch/

The ESA

www.esa.int

The SSO

http://www.sbfi.admin.ch/themen/01371/index.html?lang=en

Technologies

Qt5 documentation

http://qt-project.org/doc/qt-5/index.html

FITS 3.0 standard specification

http://fits.gsfc.nasa.gov/standard30/fits_standard30aa.pdf

ccfits website

http://heasarc.gsfc.nasa.gov/fitsio/CCfits/