# MIPP and SMHI/DMI Common Processing Environment

***Release 0.2***

**Lars Orum Rasmussen**

November 19, 2014

Contents

This is a presentation of:

CHAPTER 1

# python-mipp an introduction

`mipp` is a Meteorological Ingest-Processing Package (http://github.com/loerum/mipp).

> It's a Python libray and it's main task is to convert satellite level-1.5 data into a format understood by `mpop` (http://github.com/mraspaud/mpop).
>
> A more sophisticated interface to satellite data objects is supported by `mpop`.

In the start, it will handle **MET7**, **GEOS11**, **GOES12** and **MTSAT1R**, "eumetcasted" FSD data:

```
L-000-MTP___-MET7_________-00_7_057E-PRO______-201002261600-__
L-000-MTP___-MET7_________-00_7_057E-000001___-201002261600-C_
L-000-MTP___-MET7_________-00_7_057E-000002___-201002261600-C_
L-000-MTP___-MET7_________-00_7_057E-000003___-201002261600-C_
...
...
L-000-MSG2__-GOES11_______-00_7_135W-PRO______-201002261600-__
L-000-MSG2__-GOES11_______-00_7_135W-000001___-201002261600-C_
L-000-MSG2__-GOES11_______-00_7_135W-000002___-201002261600-C_
L-000-MSG2__-GOES11_______-00_7_135W-000003___-201002261600-C_
...
...
```

**`mipp` will:**

- decompress XRIT files (if Eumetsat's `xRITDecompress` is available).
- decode/strip-off (according to [CGMS], [MTP], [SGS]) XRIT headers and collect meta-data.
- catenate image data into a numpy-array.
  - if needed, convert 10 bit data to 16 bit
  - if a region is defined (by a slice or center, size), only read what is specified.

---

**Note:**

- MET7: not calibrated.
- GOES, METSAT: calibration constants to Kelvin or Radiance (not Reflectance).

---

## 1.1 Code Layout

**`xrit.py`**
It knows about the genric HRIT/XRIT format

```
•headers = read_headers(file_handle)
```

**`MTP.py`**
It knows about the specific format OpenMTP for MET7

- `mda = read_metadata(prologue, image_file)`

**SGS.py**
It knows about the specific format Support Ground Segments for GOES and MTSAT

- `mda = read_metadata(prologue, image_files)`

**sat.py**
It knows about satellites base on configurations files. It returns a slice-able object (see below).

- `image = load('met7', time_stamp, channel, mask=False, calibrated=True)`
- `image = load_files(prologue, image_files, **kwarg)`

**slicer.py**
It knows how to slice satellite images (return from `load(...)`). It returns meta-data and a numpy array.

- `mda, image_data = image[1300:1800,220:520]`
- `mda, image_data = image(center, size)`

**Utilities**

**cfg.py**
It knows how to read configuration files, describing satellites (see below).

**convert.py**
10 to 16 byte converter (uses a C extension)

**bin_reader.py**
It reads binary data (network byte order)

- `read_uint1(buf)`
- `read_uint2(buf)`
- `read_float4(buf)`
- ...

**mda.py**
A simple (anonymous) metadata reader and writer

**geosnav.py**
It will convert from/to pixel coordinates to/from geographical longitude, latitude coordinates.

## 1.2 Example definition of a satellite

```
# An item like:
#   name = value
# is read in python like:
#   try:
#       name = eval(value)
#   except:
#       name = str(value)
#

[satellite]
satname = 'meteosat'
number = '07'
instruments = ('mviri',)
projection = 'geos(57.0)'

[mviri-level2]
format = 'mipp'
```

```
[mviri-level1]
format = 'xrit/MTP'
dir = '/data/eumetcast/in'
filename = 'L-000-MTP___-MET7________-%(channel)s_057E-%(segment)s-%Y%m%d%H%M-__'

[mviri-1]
name = '00_7'
frequency = (0.5, 0.7, 0.9)
resolution = 2248.49
size = (5000, 5000)

[mviri-2]
name = '06_4'
frequency = (5.7, 6.4, 7.1)
resolution = 4496.98
size = (2500, 2500)

[mviri-3]
name = '11_5'
frequency = (10.5, 11.5, 12.5)
resolution = 4496.98
size = (2500, 2500)
```

## 1.3 Usage

```
import xrit

image = xrit.sat.load('meteosat07', datetime(2010, 2, 1, 10, 0), '00_7', mask=True)
mda, image_data = image(center=(50., 10.), size=(600, 500))
print mda
fname = './' + mda.product_name + '.dat'
print >>sys.stderr, 'Writing', fname
fp = open(fname, "wb")
image_data.tofile(fp)
fp.close()
```

## 1.4 A script, process_fsd

```
process_fsd --check-satellite <prologue-file>
    check if we handle this satellite

process_fsd --check [-l] <prologue-file>
    check if number of image segments are as planned
    -l, list corresponding image segment files

process_fsd --decompress [-o<output-dir>] <file> ... <file>
    decompress files to output-dir (default is working directory)
    -l, list decompressed files

process_fsd --metadata <prologue-file> <image-segment> ... <image-segment>
    print meta-data

process_fsd [-o<output-dir>] <prologue-file> <image-segment> ... <image-segment>
    it will binary dump image-data and ascii dump of meta-data)
```

CHAPTER 2

# Indices and tables

- *genindex*
- *modindex*
- *search*

[CGMS] LRIT/HRIT Global Specification; CGMS 03; Issue 2.6; 12 August 1999 “MSG Ground Segment LRIT/HRIT Mission Specific Implementation” EUM/MSG/SPE/057; Issue 6; 21 June 2006

[MTP] “The Meteosat Archive; Format Guide No. 1; Basic Imagery: OpenMTP Format”; EUM FG 1; Rev 2.1; April 2000

[SGS] “MSG Ground Segment LRIT/HRIT Mission Specific Implementation”; EUM/MSG/SPE/057; Issue 6; 21 June 2006