A Preliminary Guide to ISAP Programming

1. Introduction

1.1. What ISAP is :

ISAP is the ISO Spectral Analysis Package for the reduction and scientific analysis of ISO Spectrometer Pipeline data, and may also be used for the reduction of generic spectral line data.

1.2 . Purpose of This Manual:

The purpose of this manual is to introduce a new developer to the design and operation process of ISAP. The manual describes briefly the functionality, environment, input/ouput, structure of programming module start-up procedures and documentation.

1.3. What ISAP does:

The primary function of ISAP is to simplify the process of data handling. This includes visualizing,
shifting, rebinning, masking, combining scans with weighted means or medians, filtering, and smoothing the data. Additional analysis includes unit conversions, subsetting, line and continuum fitting, flux measurement, synthetic photometry and modeling of zodiacal light.

2. ISAP Environment

The ISAP software is written in pure IDL and is operational on Unix and VMS systems. All the required tools were designed and developed using over a time period of 5 years starting with IDL version 3.6 and ending with IDL version 5.1.4. The earlier versions were command line based where the user was invoking the procedure calls command in IDL window. The later and final version of ISAP is GUI based where ISAP startup fires a window (known as GUI WIDGET) containing buttons for certain operations. These buttons may be stand alone or the parents of other children widgets.

2.1. ISAP Directory Tree:

ISAP

- **data**
  - cal_lws - LWS calibration files
  - cal_sws - SWS calibration files
  - emissivity - emissivity tables for blackbody fit
  - filters - filter tables for Photometric measurements
  - linedb - Line data for line fit
  - web page
- **doc**
  - dcs - isap routines headers with the following filename convention
    - *.dc1 - User documentation, headers of routines which normal user can type on command line.
    - *.dc2 - Lower level IDL routines (mostly used by *.dc1) documentation.
  - perl - Perl scripts to convert dcs into html files. It also contains directory 'lib' of perl functions.
  - user_man - contains user documentation in postscript files.
  - prog_man - Postscript version of this documnet
- **pro**
  - astrolib - copies of routines used in ISAP from Astro Library
  - cesr - Line fit package originally written at CESR (and later developed at IPAC)
  - ias - Smoothing routines written at IAS
  - ipac - Routines written at IPAC
  - lia - LIA package written at RAL and IPAC (optional)
  - mpe - Routines written at MPE
  - obsolete - older version of routines which may be in command line use.
  - ral - Routines written at RAL
  - sron - Routines written at SRON
- **reduction_recipes** - Cookbook for LWS/SWS AOTS

3. ISAP Input /Output

- FITS Binary File (Final Products of Instruments Pipeline)
- FITS Ascii File (Post Mission archive survey products - Input mode only)
- ISAP saved files in IDL XDR Format (special ISAP structure MEMSTOR)
- ASCII Files containing at least 2 columns

3.1. ISAP Data Structures:

ISAP reads all input files into a IDL named structure known as ISAP AAR. This new AAR contains the following tags.

- type
  - 'LAAR' for LWS AAR Binary FITS
  - 'SAAR' for SWS AAR Binary FITS
  - 'YAAAR' for other types of input Files and ISAP processed AAR
- header
  - Array of strings of 80 chars
    Fits Files : Fits primary headers are copied into header. It also contains the secondary headers reconstructed using the data tag names.
    Ascii Files: Inserts a FITS like primary header with zero or blank values and the secondary header using the data tag names.
- history
  - Array of strings of 32700 chars each
    Contains all 'HISTORY' keywords found in input Fits header
- data - IDL structure containing the following possible tags:

  {WAVE, WAVEU, FLUX, STDEV, TINT, DET, ITK, UTK, RPID,
  SPARE, LINE, SDIR, SCNT, STATUS, FLAG}

The FITS primary and secondary headers are written into the header tag of AAR. In addition,
all HISTORY keywords found in FITS headers are saved in history tag.

The translation of input FITS file parameters into ISAP AAR data tags for each instrument is given as follows:
For a description of SWS AAR files, please see ISO-SWS Instrument & Data Manual.

SWS : Fits Parameters [SWAAWAVE, SWAAFLUX, SWAASTDV, SWAATINT, SWAADETN, SWAAITK, SWAAUTK, SWAARPID,SWAASPAR , SWAALINE , SWAASDIR, SWAASCNT, SWAASTAT, SWAAFLAG] are translated into AAR data tags [WAVE, FLUX, STDEV, TINT, DET, ITK , UTK , RPID, SPARE, LINE , SDIR, SCNT, STATUS, FLAG].

For a description of LWS AAR files, please see ISOLWS Data Users Manual.

LWS: Fits Parameters [LSANWAV, LSANWAVU, LSANFLX, LSANFLXU, LSANTINT, LSANDET, LSANITK, LSANUTK, LSANRPID,
LSANFILL, LSANLINE, LSANSDIR, LSANSCNT, LSANSTAT, LSANFLAG] are translated into AAR data tags
[WAVE, WAVEU, FLUX, STDEV, TINT, DET, ITK, UTK, RPID, SPARE, LINE, SDIR, SCNT, STATUS, FLAG]

PHT: All PHT-S Fits files have data records with each record representing a single raster point. Each data record also have more than 1 type of flux (source, source + background etc.). Individual AARs are created for each type of flux (user chosen) containing the data tags [WAVE, FLUX, STDEV, FLAG] where WAVE is obtained from keywords LAMBDnnn, nnn=1, 64 and FLUX and STDEV is the flux values and their uncertainties.

CAM CVF: CAM CVF Fits files have data records with each record representing a single pixel in CAM array. Each data record also have more than 1 type of flux (source, source + background etc.). Individual AARs are created for each type of flux (user chosen) containing the data tags [WAVE, FLUX, STDEV, FLAG] where WAVE is obtained from keywords LAMBDnnn, nnn=1, 64 and FLUX and STDEV is the flux values and their uncertainties.

YAAAR:

Input to ISAP are the files as described above. However, sometime during processing information is lost on certain data tags. For example when one rebins/merges/averages spectra from individual detectors into one final spectrum, one looses the information on individual detectors. Thus, an ISAP processed AAR , known as YAAAR (Yet Another AAR), can have 4 to 15 data tags. The minimum data tags expected are WAVE, FLUX, STDEV, and FLAG. Also, non-ISO Ascii data are read in as YAAAR.

4. ISAP Coding Conventions

All modules were designed with the following concepts in mind.

- The top gui module and its event handler are contained in top level gui module DATA_HANDLER8. From here all ISAP functions are accessible via buttons.
- Each button is used by writing its own gui module (name starting with 'gui_') which also contains its event handler.
- Each button application is written in a separate module(s) called by the gui event handler.
- The data transfer between gui and event handler is achieved using top widget button user value.
- The data transfer between modules is achieved using data handles (apart from the passing arguments)
- The graphic display (and its controls) in each widget is obtained using a Global Routine CW_DRAW.
- Use of common routines (Appendix A) and common features wherever possible

- Each routine contains documentation in the form of ISAP Standard Header (Appendix C)

4.1. ISAP Flow:

All the top level functions and procedures containing the major functionality and called by data_handler.pro are identified within parenthesis [] . The lower level routines are identified under 'Dependencies' in the headers of top level routines.

- **IDL Start Script:**
  This script defines the directory paths to all the required files , defines system variables and starts the main gui **[sci_anal5]** which initiates the event handler **[data_handler8].** Once in data_handler8, the following events can be performed.

- **Read Input Data** in either of the following form : Each input file is converted into an AAR structure.
  - AAR Fits Files - The IDL procedure loadstuff handles the following types. In case of raster data, the user is given a choice of selecting rasters [**check_raster**] and a data set is generated and stored for each selected raster.
    - CAM CVF **[sap_rfitccv]** - one data set for each raster point is read and stored.
    - SWS/LWS/YAAAR **[sap_rfits]** - Bad points are removed from the input data set before storing.
    - PHT-S **[sap_rfitsp]** - one data set for each raster point and for each selected type of flux **[gui_rfitsp]**
  - LWS Filter Function **[sap_rfit_lcgr]**
  - SWS calibration file **[sap_rfit_sws]**
  - AAR XDR - convert IDL XDR formatted files into AAR structure **[data_handler8]**
  - AAR ASCII - converts ASCII files containing at least 2 columns of data into AAR **ascii2yaaar]**
  - Survey Products - converts ISO Archive Browse Fits Ascii files into AAR **[loadsp, sap_rspfits]**
- **Store it [local_store]** - Once the input file is converted into AAR structure, an IDL dynamically allocated variable identified by a unique integer value (known as data handle) is created for this AAR and saved in the memory. These handles (identified by the input file names) can be displayed in 'Stored Data Sets' window **[stored_ds)** and any data set can be retrieved at any time in the session by selecting its handle in the window. The selected handle value (AAR structure) is loaded into memory and the user has a choice of applying different functions to this AAR (make it prime, use as REF, display etc.)

- **Make it Prime data [make_prime]** The Prime data is the one whose tags (det, scan, lines, sdir, bands) information are displayed in the upper half of ISAP Gui and is used for all processing. In general, the most recently read data is the prime data., or once the data is stored, it can be retrieved by selecting its handle in 'Stored Data Sets' which loads the selected handle value into AAR structure and making it Prime data set.
- **Alteration of Input Data** - This is done using the menu button 'Special Fnc..' Change Units **[gui_units_conv,]** - units conversion for wave, flux, and stdev. The acceptable units in isap are

  wave : {'m','cm',','microns',um','a','hz','1/cm'} where a = Angstrom. **[wave_form]**

  flux: {'jy', 'w/cm^2/um', 'w/m^2/um', 'w/cm^2/hz','erg/s/cm^2/um', 'erg/s/m^2/um', 'erg/s/cm^2/hz']. '/sr' is allowed for all flux units. **[spect_form]**
  - Add data tag **[gui_add_tag , add_tag]** - add a new tag to aar data from the allowable list {WAVEU, TINT, DET, ITK, UTK, RPID, LINE, SDIR, SCNT, STATUS}.
  - Increment data tag **[gui_increment_tag , increment tag]** - increment data tags {det,line,scan,sdir} by the user defined values.
  - Fix SWS07 **[data_handler8]**
  - Remove bad LWS data **[debum_aar]**
  - Remove ALL bad SWS data **[remove_data]**
  - Convert LWS Fabry Perot Flux Unit **[lws_fp_res]** -convert LWS FP fluxes into w/cm^2/um
  - Apply Extended Source Correction (LWS)
  - Defringe LWS AOT1 **[cl_fringe2]**
  - Defringe (SWS Algorithm) **[sws_defringe]**
  - Convert Line -> AOT Band(SWS)
  - Split Apart AAR **[split_aar]**
  - Merge group of AARS
- **Data Display -** For the user selected tags (det, scan, line, sdir), data can be displayed as follows:
  - visual display using 'DispAAR' **[disp_aar]** under Special Fnc...
  - in CW_DRAW window **[plot_aar]** in which data is selected by making a box using right mouse button and the application is selected.
- **Preselection -** The input data may also be reduced by applying specific basic functions (given below) under this button. e.g. selecting and masking different portions of data , store it, and use it as prime data.
- **Applications** - Various applications can be performed on selected data (see CW_DRAW). They can be grouped as follows.
  - Applications providing basic tools contains

- Fix Stdev **[gui_fix_stdev]**
- Make AAR/Reference **[data_handler8, redo_hdr]**
- Mask/Unmask **[mask]**
- Normalize **[norm]**
- Oplot to ref **[gui_oplot]**
- Selection info **[show_info]**
- Shift **[sap_shift]**
- Tippex **[tippex]**
- Xshift **[gui_xshift]**
- Zap/Zap masked **[analyse2, extrcat_selected4]**

- Applications driven by **FEEDER/MERGER [gui_feeder, feeder, merger].** They are Average **[gui_average),** Arithm2Ref **(gui_arithm2),** Clipping **[gui_average],** Rebin(**gui_rebin4),** Rebin2Ref **(gui_rebin4),** Smooth [**gui_smooth],** Split_aar **(split_aar).** The common programming features among all these applications are as follows :
  - check if four tags (wave, flux, stdev, flag) are present. if not, terminate the operation.
  - remove the masked data .
  - check which of the four tags (scan, det, line, sdir) are present and multi valued. For these tags found, build a sub widget in the top left corner of application GUI containing the buttons (on/off)> for these tags.
  - If the tag is selected, the unique portion of data for that tag is extracted from the input data and operation is performed on it. Thus, if no tags are selected, then the whole aar will be processed.
  - All applications routines pass the data in/out arrays.
  - Tags selected above will be saved in output AAR in addition to tags (wave, flux, stdev, flag)..
  - Merge back the masked data to output AAR.
  - Exceptions to above rules are
Keep masked data during Arithm2Ref operation.
save RPID tag in output AARS for rasters
redefine STATUS tag containing the active det only for LWS AOTS.
- More advanced science applications consist of :
  - Photometry **[gui_syn_phot]**: to provide photometry measurements of IS0 SWS and LWS spectra or other spectra
  - Line Fit **[gui_fit]**: to fit models to flux continuum and spectral line data.
  - Blackbody **[gui_bbfit]**: to fit one, two, or three summed blackbody components to a given AAR.
  - Zodiacal light **[gui_zodiacal]**: to compute a model of zodiacal emission for the wavelengths in a given AAR.

5. Documentation

Scripts have been written in Perl (upto version 5.004) on Unix system to create automated documentation for ISAP. They are briefly described as follows.

5.1. Header Extraction:

The headers of all the routines (* .pro) in a directory can be extracted using the perl scripts xfile.pl and extract_file.pl (in ./doc/perl) using the following command.

```
setenv PERLLIB  lib
alias xfile 'perl xfile.pl'
xfile -v -p  output_directory  input_directory/*.pro
```

The output_dirctory will contain the files with .dc1,.dc2 extensions.

5.2 . HTML Conversion:

Define the environment variable ISAP_DOC pointing to the directory containing header files (*.dc) .

```
setenv  ISAP_DOC   output_directory
```

Execute Perl script 'make_isap_html_libs' to create html files and index files in the current directory.

A. List of Common Routines Written in ISAP:

ACKNOWLEDGE - Error handling for GUI mode

BANDS_TO_DETS - to convert SWS bands to the list of detectors

CHK_TAG - to check if a given AAR contains a particular tag

COMBINE - to combine two AARS data structures

CUT_AAR - to obtain specific data records from an AAR

CW_DRAW - to create ISAP Plot Windows and their events. This routine defines the actions of mouse buttons which are

- Data selection : drag using right mouse button. will pop up the application selection widget for the selected data.

- Zooming : drag using left mouse button
- Data info : shown in message window for the point selected by center mouse button

FEEDER - to perform application on unique parts (or whole) of an AAR

FIX_SCALAR - to **c**onvert a scalar to 1 element array

GET_HDRKEY - to read a keyword value in FITS file header

GET_AOTB_INDEX - to get the AOT number of the instrument from FITS file header

GET_UNITS - to obtain wave and flux units from FITS header

GUI_FEEDER - to select different parts of an AAR

GUI_PLOTSTYLE - to set the plot styles (color,symbols etc.) for cw_draw window plots

IS_STRUCT - to check if input AAR is legal

KWD_NOTSET - to check the keyword setting and its legality

LOCAL_STORE - to store a data set for later use

MAKE_PRIME - to make a data set Prime (current)

MERGER - to put masked data back into output AAR (applications driven by FEEDER)

MERGETWO - to combine two similar data structures,

PLOT_ALL - for plotting in CW_DRAW windows,

RESET_DATE - to convert incoming dates in input files to Y2K compliance dates

SAP_ERROR - Error handling for command line and GUI modes

SAP_WFITS - to write an Fits Binary file from an AAR structure

SET_USERSYM - to set user defined symbols generated with symgen

SHRINK_ARRAY - to determine the unique values of a data tag

SYMGEN - to generate vectors for user defined symbols

UPDATE_HISTORY - to update the history tag in AAR

WRITEASC - to write an AAR data into an Ascii file

WRITE2HIST - to write a string into 'history' tag of an AAR

WRITE_HDRKEY - to write a keyword into "header" tag of an AAR (which is a copy of FITS header)

B. List of Common features Used in ISAP Routines:

1. Error status via routine argument 'err_stat' which is defined 0 for normal, 1 for error.
2. Messages /Info written in ISAP message window (at the bottom) by passing the id of message text widget in routines argument .

C. ISAP Standard Header:

;#> Identifier.dcX

; Identifier
; Purpose
; Synopsis
; Arguments Name I/O Type Description

; Returns
; Description
; Comment

; Example
; Dependencies CALLS:
; CALLED BY:
; Category
; Filename
; Author
; Version
; History

; Copyright
;#<

D. An ISAP Coding Example:

Below is given an example to add a new routine to ISAP. The objective is to display a plot of
wave vs flam (wave*flux) for the prime AAR (data displayed in main ISAP plot window). A new
button is added to perform this operation. This button can be added at two possible places.

1. Either in tool kit window which appears as soon as the data is selected with right mouse button. The routine "make_selection.pro" will be changed for this purpose. The following instruction is added at line # 154.

```
junk = widget_button(rebbase,value = 'Plot FLAM',uval = {type:'plot_flam'})
```

This will make a button appear in tool kit window with label "Plot FLAM", and the type of the event is 'plot_flam'. Next, the ISAP event handler will be changed to perform the actual operation. Please note that the event handler contained in "data_handler8.pro" consists of 2 parts. The first part consists of the events handling generated by CW_DRAW select mode (which is the above case). The following lines are added to this part (at line #1492)

```
'plot_flam' : begin
        handle_value,info.current_data,cdata
         msg='plot wave vs flam ...'
         widget_control,info.text_wid,set_value = msg ,/append
         stat = gui_plot_flam( select_aar, info.pstyle_op1, $       ;selected aar only
                       title=cdata.name)
      end
```

Also note that the function is performed on the selected part of data only.

2. The button can also be placed under "Special Funcs.." which is a menu button on ISAP main Gui. In this case, the function will be performed on the whole AAR. The following line is added to the data_handler8.pro at line #2852

```
plflam   = widget_button(spec_butt,value = 'Plot FLAM',uval = {type:'plot_flam'})
```

This will make a button with label 'Plot FLAM'appear under Special Funcs menu. The following lines will be inserted to the second part (line# 1738) of event handler in data_handler8.pro which handles events depending on the event user value .

```
'plot_flam' : begin
        handle_value,info.current_data,cdata
```

```
    msg='plot wave vs flam ...'
    widget_control,info.text_wid,set_value = msg ,/append
    stat = gui_plot_flam( cdata.aar , info.pstyle_op1, $     ;whole aar
             title=cdata.name)
  end
```

The functional routine gui_plot_flam.pro used in both above cases by event handler is listed below.

```
;#> gui_plot_flam.dc2
;
; Identifier    gui_plot_flam
;
; Purpose       to plot lambda vs flam
;
;
; Synopsis      status = gui_plot_flam,aar,      ; the input aar
;                           pstyle,              ; plot styles
;                           title = title,       ; optional title
;                           group = group)       ; group number
;
; Arguments     Name    i/o   type            description
;               ---------------------------------------------------------
;               aar       I    int         aar
;               pstyle    I    int         handle_id to array containing
;                                          plot style info (7 elem array)
;                                          (symbol,color,line specs etc)
;               title     I    string      title of the cw_draw
;               group     I    int         group leader number
;
; Returns       status, 0 for normal, 1 otherwise
;
; Description
;
; Dependencies  Calls: ACKNOWLEDGE, GET_HDRKEY, KWD_NOTSET, PLOT_ALL,
;                      SAP_ERROR
;               Called By: DATA_HANDLER8
;
; Category      ISAP
;
; Filename      gui_plot_flam.pro
;
; Author        authors's name
;
; Version       1.0
;
; History
;
;*******************************************************************
;#<

pro gui_plot_flam_event,ev                       ; Event Handler of plot_flam

widget_control,/hourglass                        ; tell em to wait
widget_control,ev.id, get_uval = uv
type = tag_names(ev,/struct)

if type eq 'CW_DRAW' then return                 ;zoom, select, redraw etc
```

```
CASE uv.type OF
   'quit'    : widget_control,ev.top,/destroy   ; destroy everything
   else      : print,uv.type
ENDCASE
END

;***************************************************************
FUNCTION gui_plot_flam,aar,pstyle,title = title, $
                   text_wid=text_wid, group=group

IF KWD_NOTSET(title,"title", "str") THEN  $
   title = 'PLOT FLAM'

IF KWD_NOTSET(text_wid, "text_wid", "int") THEN $
    text_wid = -1           ;do not write info text in isap msg window

;check for flux units in AAR header
;only 'w/cm^2/um' for flux are acceptable
 header = aar.header
 fstat = GET_HDRKEY(header,'TUNIT2',funit, stat=stat)

 IF fstat EQ 1  THEN BEGIN
   SAP_ERROR,'F', $
    'Flux unit definition (TUNIT2) does not exist in AAR header'
   return, 1

 ENDIF ELSE IF STRPOS(funit, 'w/cm^2/um') LT 0  THEN BEGIN
   funit = STRLOWCASE(funit)
   SAP_ERROR, 'F', $
     "Please change flux units " + funit + " to w/cm^2/um"
   return, 1
 ENDIF

main_base = widget_base(/column)            ;top base for this Gui
cw1 = CW_DRAW(main_base,$                   ;plot window in top base
                 title = title,$
                 xsize=650,$
                 ysize=400,$
                 max_oplots = 2000,$
                 xmargin = [15,3],$
                 ymargin = [5,2])

junk = widget_button(main_base, $           ;Quit button
        value = 'Quit',uval = {type:'quit'})

;set up the group leader, if desired
;this widget will die when group leader is killed
IF NOT KWD_NOTSET(group, "group","int") THEN BEGIN      ; if group defined,
   IF widget_info(group,/valid_id) eq 1 THEN BEGIN      ; check it & set it up
       widget_control,main_base,group_leader = group
   ENDIF ELSE BEGIN
      ACKNOWLEDGE,text = 'PLOT_FLAM: Invalid Group Leader Id : ' $
                         + strtrim(group)
   ENDELSE
ENDIF

tmp = aar
tmp.data.flux = aar.data.wave*aar.data.flux     ;flam = wave*flux
ytit = "flam (Watts/cm^2)"                      ;redefine output units

widget_control,main_base,/realize               ; realize it
```

```
PLOT_ALL,tmp,$                                      ; plot the converted data
         cw1,$
         pstyle, $
         textwid=text_wid, $
         title = title, $
         ytitle = ytit

xmanager,'gui_plot_flam',main_base                  ; start event handler
                                                    ; gui_plot_flam_event
                                                    ; the top procedure


RETURN, 0                                           ;normal status
END
```

E. ISAP Designers and Programmers

ISAP has mainly been developed by members the SWS and LWS Instrument teams and the ISO Data Centers at the following institutions. A table of developers at each institute also follows.

- CESR: Centre d'Etude Spatiale des Rayonnements
- IAS : Institut d'Astrophysique Spatiale, Orsay, France
- IPAC: Infrared Processing and Analysis Center, Pasadena, CA, USA
- MPE : Max-Planck-Institut fuer extraterrestrische Physik, Garching, Germany.
- RAL : Rutherford Appleton Lab, UK
- SRON: Space Research Organization Netherlands, Groningen

ISAPer	Institute
Otto Bauer	MPE
Jim Brauher	IPAC
Mark Buckley	RAL
Andrew Harwood	RAL
Min Hur	IPAC
Iffat Khan	IPAC
Jing Li	IPAC
Steve Lord	IPAC
Dieter Lutz	MPE
Joe Mazzarella	IPAC
Sergio Molinari	IPAC
Pat Morris	SRON
Bob Narron	IPAC
Karla Seidenschwang	MPE
Sunil Sidher	RAL
Eckhard Sturm	MPE
Bruce Swinyard	RAL
Sarah Unger	QMW
Laurent Verstraete	IAS
Florebce Vivares	CESR
Ecki Wieprecht	MPE

Last update: 26-April-2000