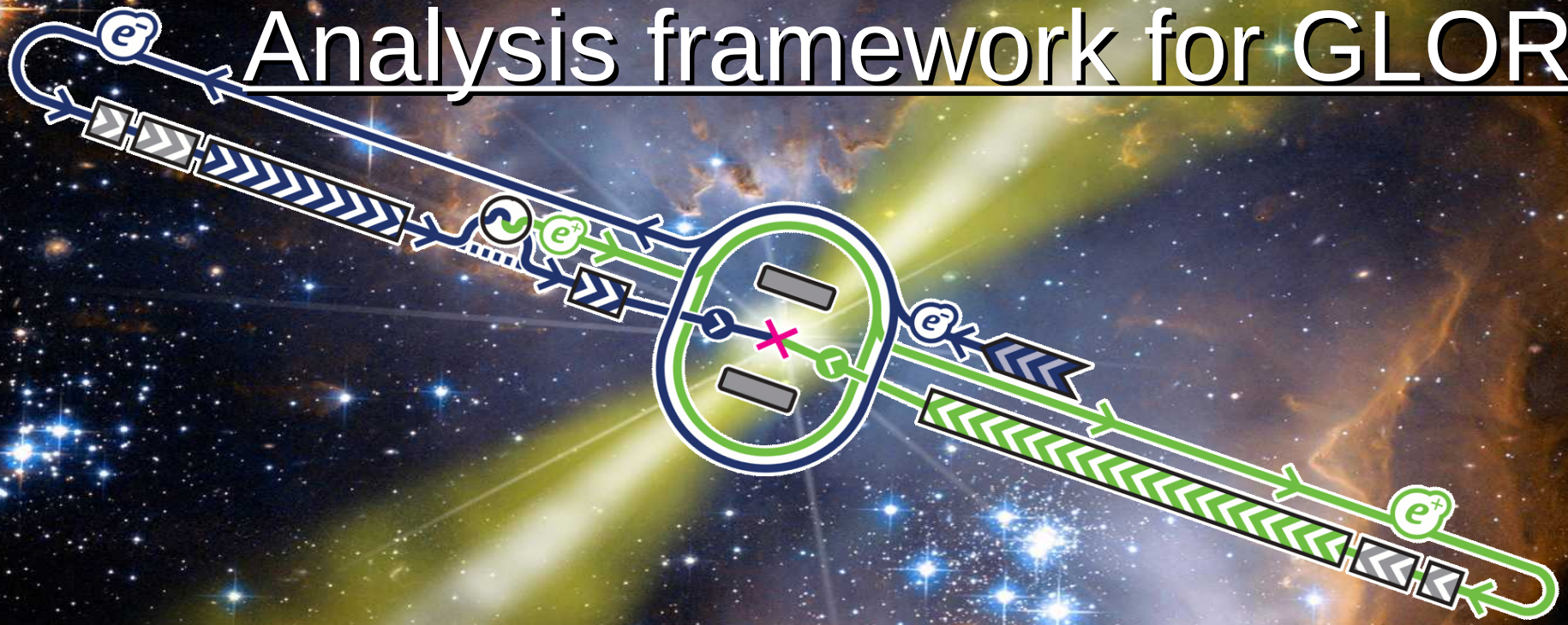
LUIZA

Analysis framework for GLORIA



Aleksander Filip Żarnecki
zarnecki@fuw.edu.pl
University of Warsaw

XXX-th IEEE-SPIE Joint Symposium Wilga 2012

Outline

- GLORIA project
- Basic concept
- Luiza framework
- First analysis tools
- Conclusions

GLORIA

- FP7 project

http://gloria-project.eu/

GLObal Robotic telescopes Intelligent Array for e-Science

TYPE OF FUNDING SCHEME:

COMBINATION OF COLLABORATIVE PROJECTS AND COORDINATION AND SUPPORT ACTIONS (CP-CSA)

WORK PROGRAMME TOPICS ADDRESSED:

INFRA-2011-1.2.1: e-Science environments

GLORIA

- 13 partners, 17 robotic telescopes

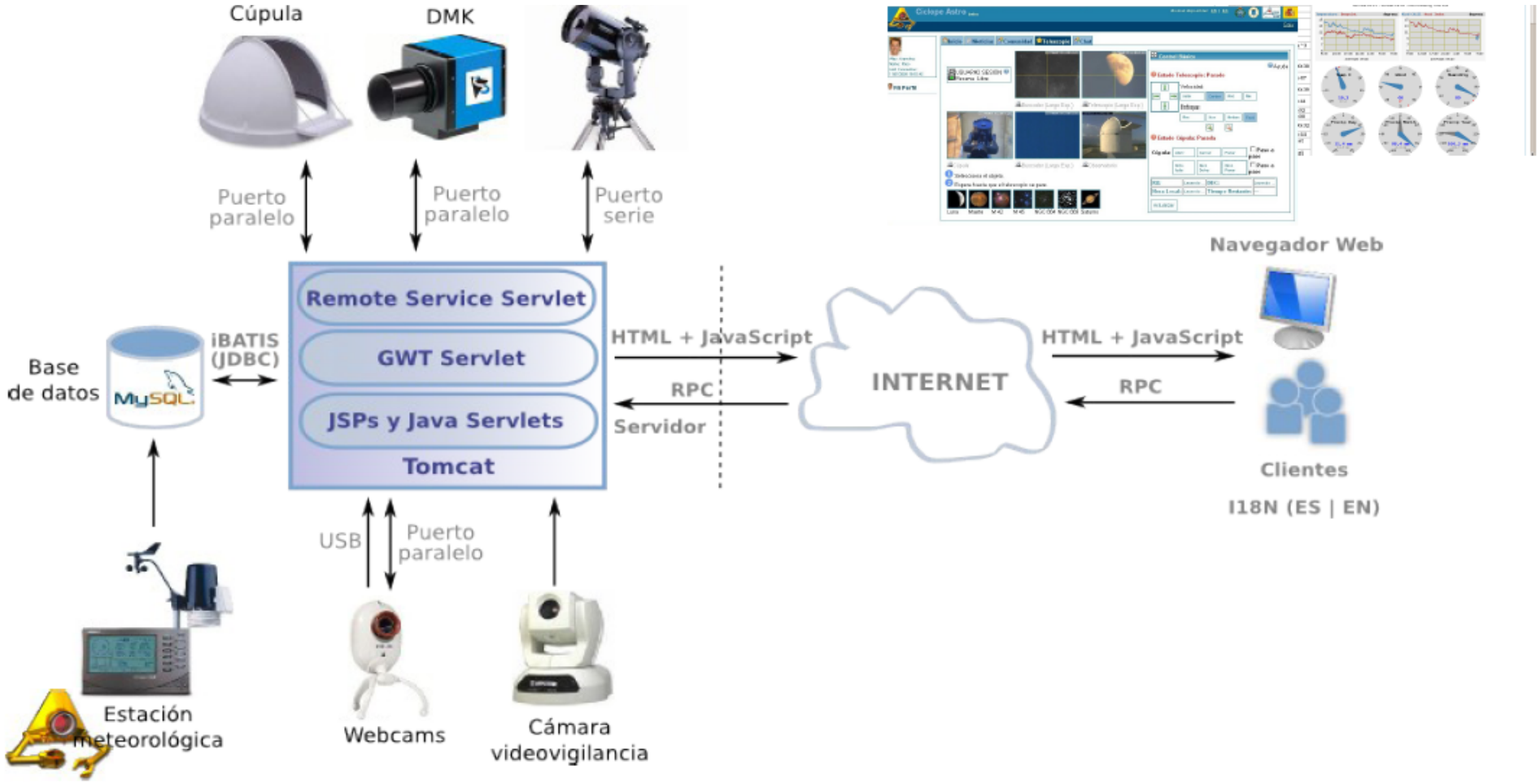
GLORIA

- Project goals:
  - Create free and open-access network of robotic telescopes for citizen science
  - Develop Web 2.0 environment for easy access to telescopes and other network resources

GLORIA

- Approach based on the Ciclope project

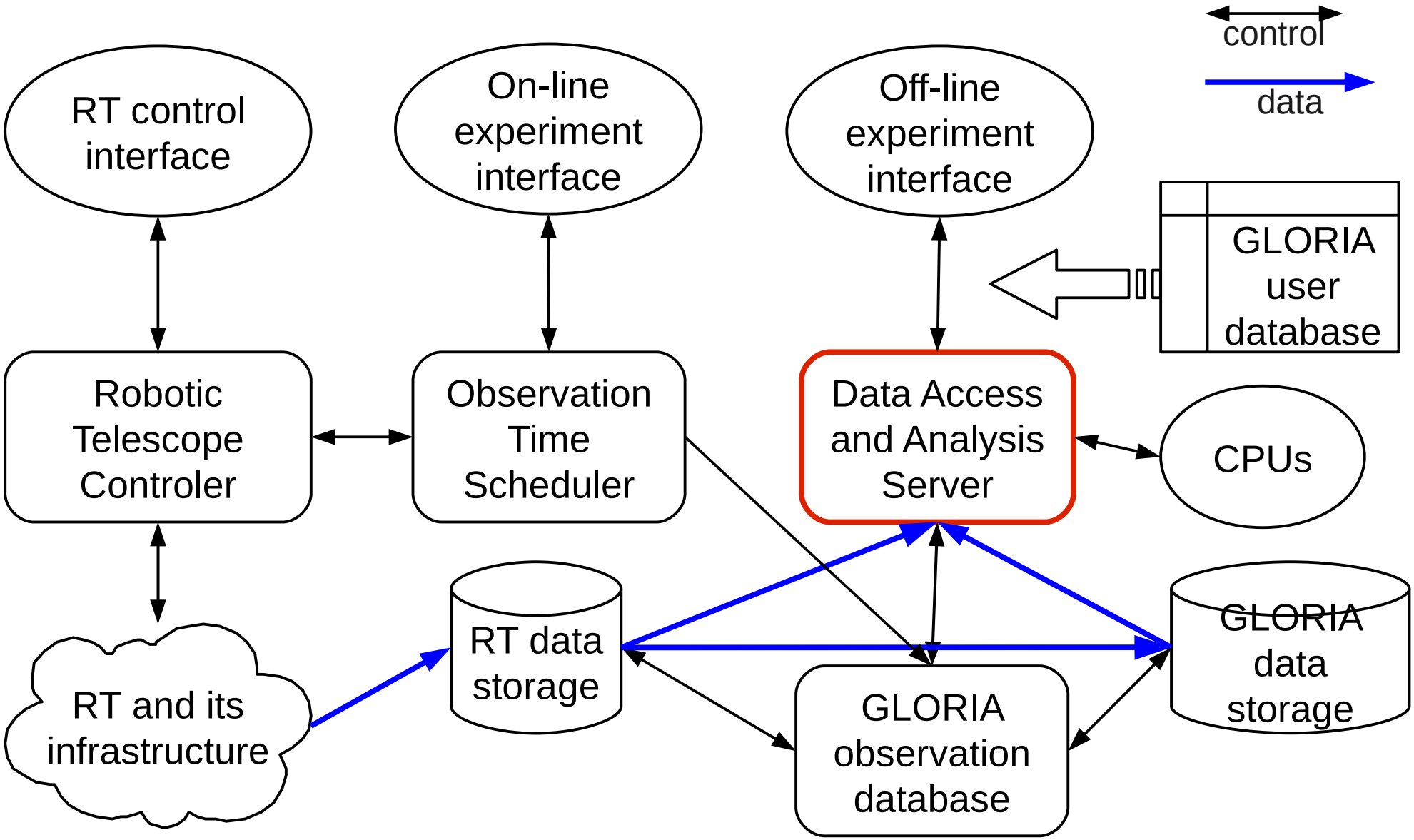
  Montegancedo Astronomical Observatory, UPM, Madryt



GLORIA

- Project goals:
  - Create free and open-access network of robotic telescopes for citizen science
  - Develop Web 2.0 environment for easy access to telescopes and other network resources
  - Design tools for doing „on-line” experiments (observations) and **„off-line” analysis**
    new quality: observations possible 24/24
  - Develop full framework for doing research with robotic telescopes, allowing also for easy integration with GLORIA network
  - Outreach in science and astronomy

GLORIA structure



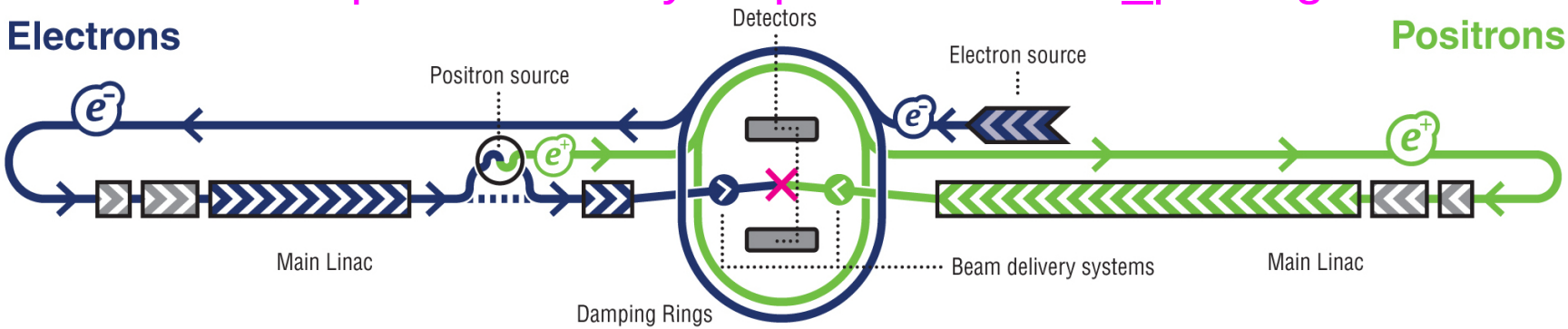
Basic concept

- High Energy Physics experiments deal with enormous amounts of data
  - Experiments at LHC take “images” every 50 ns, each “image” is ~100 Mpixel (electronics channels)
  - Even after very strong (10^{-5}) on-line selection GBs of data are stored every second
  - Data analysis on LHC Computing Grid (WLCG)
    - about 170 000 TB of disk space
    - CPU power of about 1 800 000 units (HEP-SPEC06)

Basic concept

- International Linear Collider (ILC) project
  Next generation e^+e^- collider under study (2020?)
  - Large amount of future data expected
  - Large samples of Monte Carlo data already generated to test detector performance and analysis methods
  - **Marlin framework** developed for efficient data reconstruction ("reduction") and analysis

http://ilcsoft.desy.de/portal/software_packages/marlin/



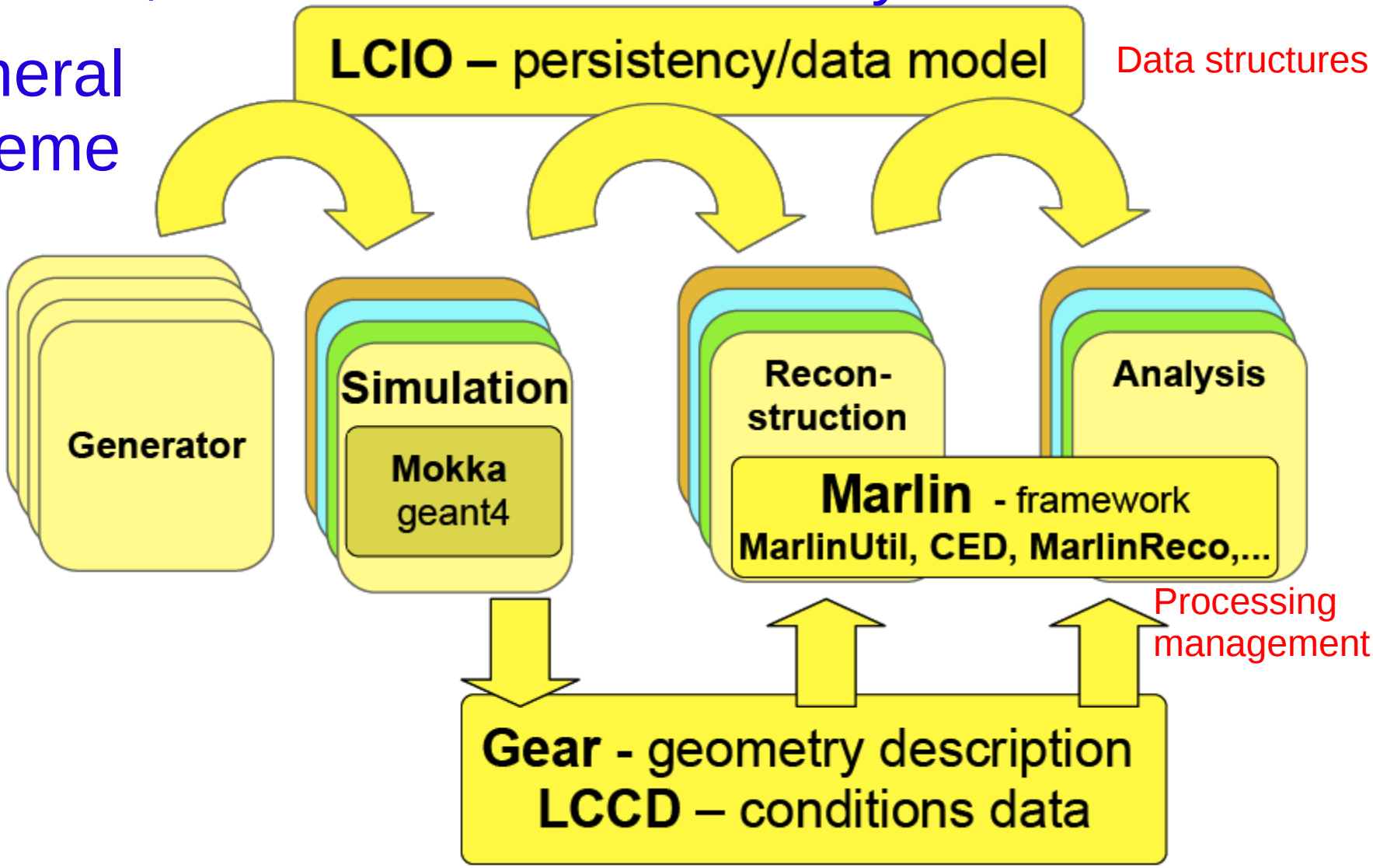
Basic concept

- **Marlin** is based on the following approach:
  - Data reconstruction and analysis should be divided into small, well defined steps (implemented as so called **processors**)
  - Each step has to have well defined input and output data structure
  - By defining universal data structures we make sure that different processors can be connected in a single analysis chain, i.e. exchange data and analysis results
  - Processor configuration and their parameters can be set by user at run time in a simple steering file
- Same concept used in **Luiza framework**

Basic concept

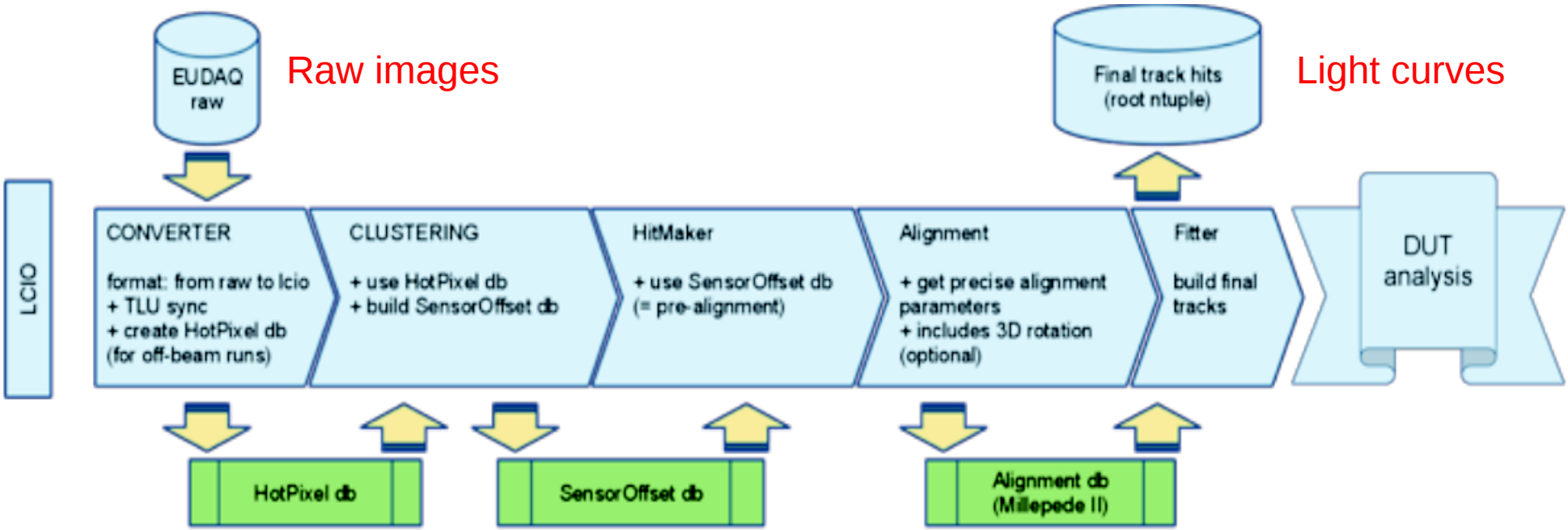
Simulation, reconstruction and analysis for ILC

- General scheme



Basic concept

- Example analysis chain for silicon pixel detector
  (similar to CCD, but measures charged particles instead of photons)



- Starting from “reduction” of raw “images” sophisticated analysis can be made...

Luiza framework

Data structures

- GloriaFitsImage - class for storing FITS images
  - uses fitsio library for reading and storing images
  - basic methods for image manipulation

all other data can be stored in

- GloriaFitsTable - class for storing data
  - flexible (integers, floats, strings, vectors of int/float)

both classes use

- GloriaFitsHeader - class for FITS header data

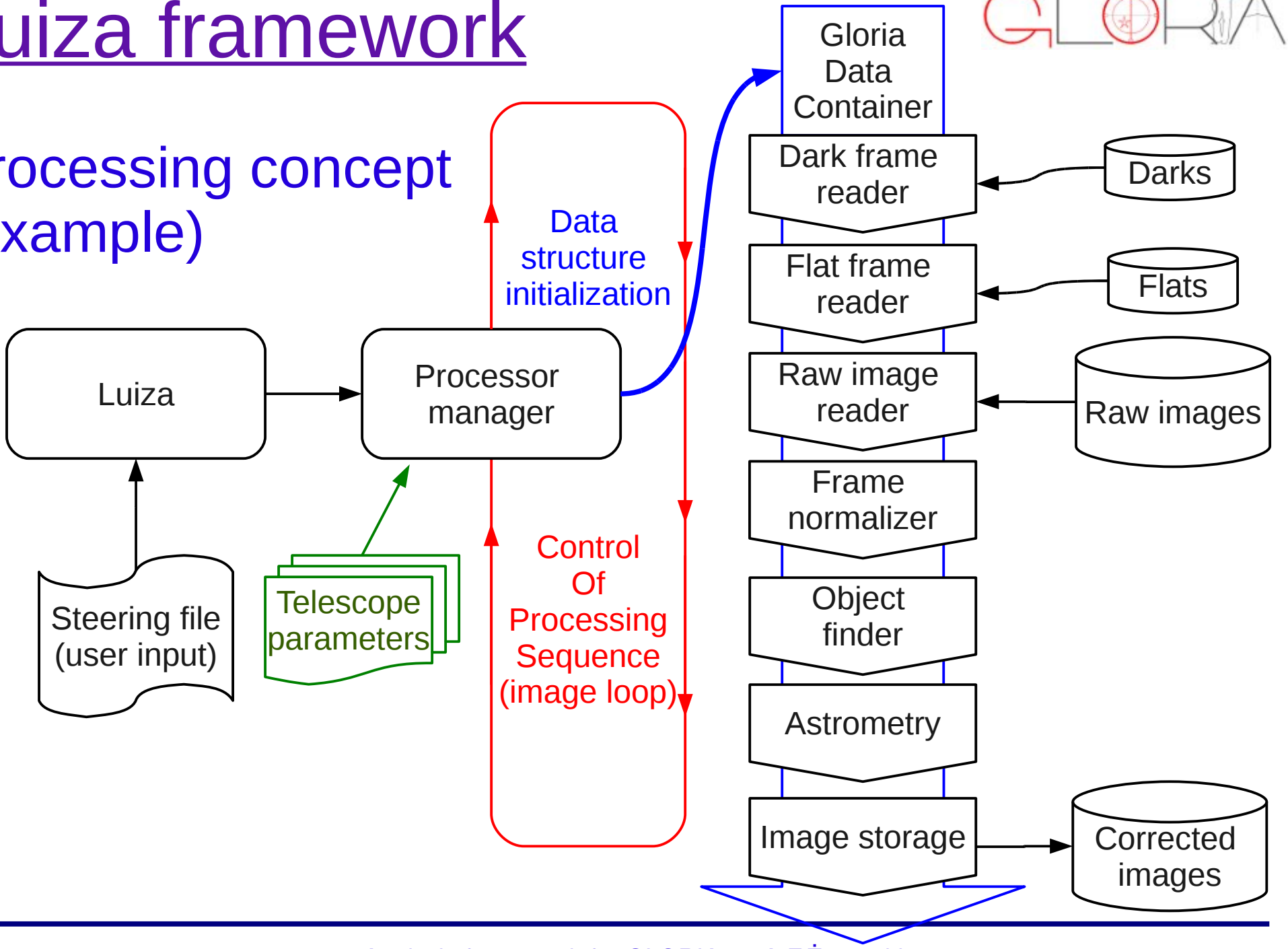
Luiza framework

- GloriaDataContainer – internal storage class
  - Images and tables stored in “collections”
  - Each collection has a unique name (string)
  - Each can include multiple images or tables (vectors)

- Each Luiza processor gets a pointer to global GloriaDataContainer
  - Can analyse data already stored in memory
  - Can add new collections
    (data read from storage or analysis results)

Luiza framework

Processing concept (example)



Luiza framework

- Steering file
  - Allows user to select processors and their order

    Example (corresponding to the scheme shown):

```
<execute>
  <processor name="DarkImageReader"/>
  <processor name="FlatImageReader"/>
  <processor name="RawImageReader"/>
  <processor name="CalibrateImage"/>
  <processor name="FindObjects"/>
  <processor name="DoAstrometry"/>
  <processor name="StoreFinalImage"/>
 </execute>
```

Luiza framework

- Steering file
  - Sets parameters for each processor

    Example (darks reading):

```
<processor name="DarkImageReader" type="FitsImageReader">
  <!--Processor for reading input FITS images. Reads images from given files-->
  <!--List of FITS files to be read-->
  <parameter name="FitsFileList" type="StringVec"> dark.fit  </parameter>
  <!--Name of the image collection to which images from file should be stored-->
  <parameter name="ImageCollectionName" type="string">DarkImages </parameter>
  <!--Number of images to be read per processing loop (0 for all))-->
  <parameter name="ImagesPerLoop" type="int">0 </parameter>
  <!--Name of file containing FITS file list (one per line)-->
  <parameter name="ListFileName" type="string"> </parameter>
  <!--Flag for collections, which should not be deleted after loop is finished-->
  <parameter name="PermanentCollection" type="bool">true </parameter>
  <!--verbosity level for processor ("DEBUG,MESSAGE,WARNING,ERROR,SILENT")-->
  <parameter name="Verbosity" type="string"> MESSAGE </parameter>
</processor>
```

First analysis tools

- So far development focused on general structure and functionality
  - Data structures implemented
  - Steer file parsing and processing management adopted from Marlin
  - Input/output processors
    - Input and output of FITS image files (using fitsio)
    - Output of tables to text file
    - Image viewer based on CERN root package
    - LuizaGUI for creating/editing steering files

First analysis tools

- First processors allowing for simple image analysis (and framework tests):
  - Image normalization (dark subtraction, flat correction)
  - Image stacking (or averaging)
  - Object finding
    Two simple algorithms implemented
    - PixelClusterFinder – based on the particle identification algorithm developed for silicon pixel detectors
    - SimpleSourceFinder – based on Python library *Mahotas*
  - Astrometry algorithm based on Astrometry.net (tests in progress)

Development plans

- We work on implementing basic tools for image analysis (astrometry, photometry, light curve)
  - General algorithms are never the most precise ones
    ⇒ can be used as examples
    and starting point for future improvements
- Further development as a part of GLORIA experiments and by individual users
  - New packages can be compiled as independent libraries, loaded dynamically at run time
    ⇒ user can compile and load only what is needed
    ⇒ user can develop and link “private” code
    (can be later included in Luiza as a separate package)

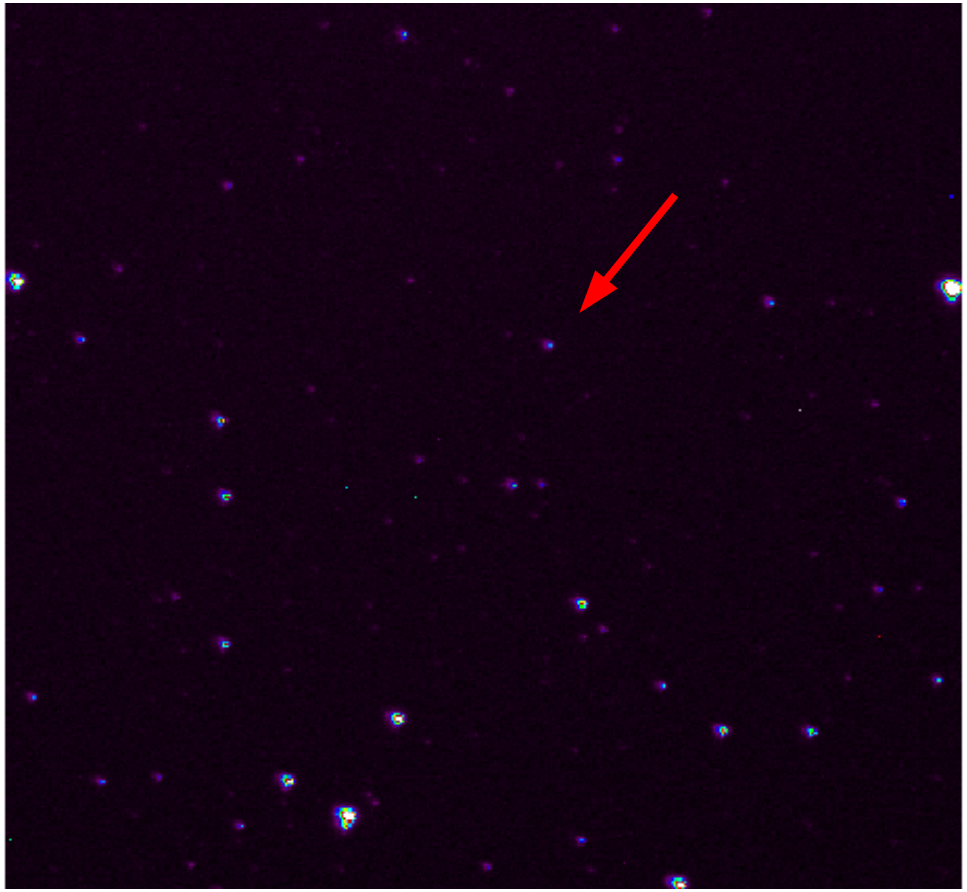
Development plans

- Processors for GLORIA experiments (ideas)
  - Interface to star catalogues, external databases
  - Interface to Virtual Observatory resources
  - Smart image stacking (with shifts, rotations, cuts)
  - Frame quality checks
  - High quality photometry (aperture and profile)
  - Light curve determination and variability analysis
  - Search for optical bursts, flares etc.

many more possibilities...

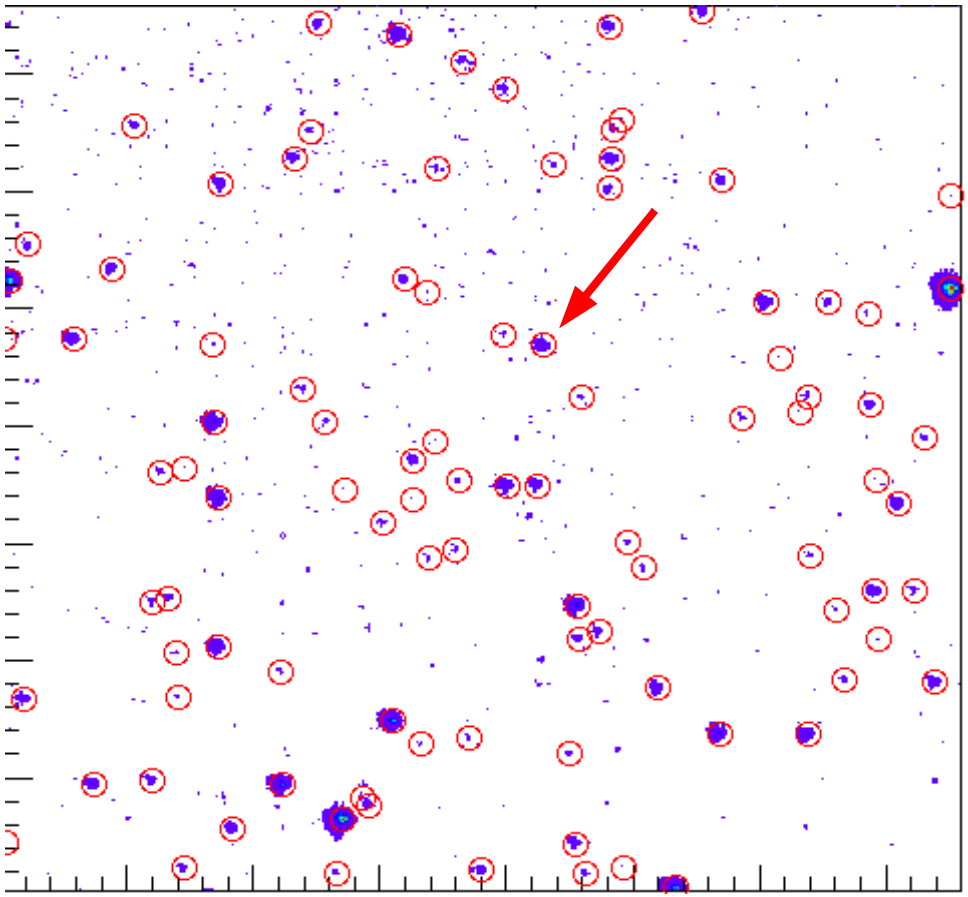
Working example

- Using Bootes-1 data taken on Nov. 25, 2011 (thanks to Martin Jelinek and Petr Kubanek)
- Processing procedure
  - Image reading
  - Image display



Working example

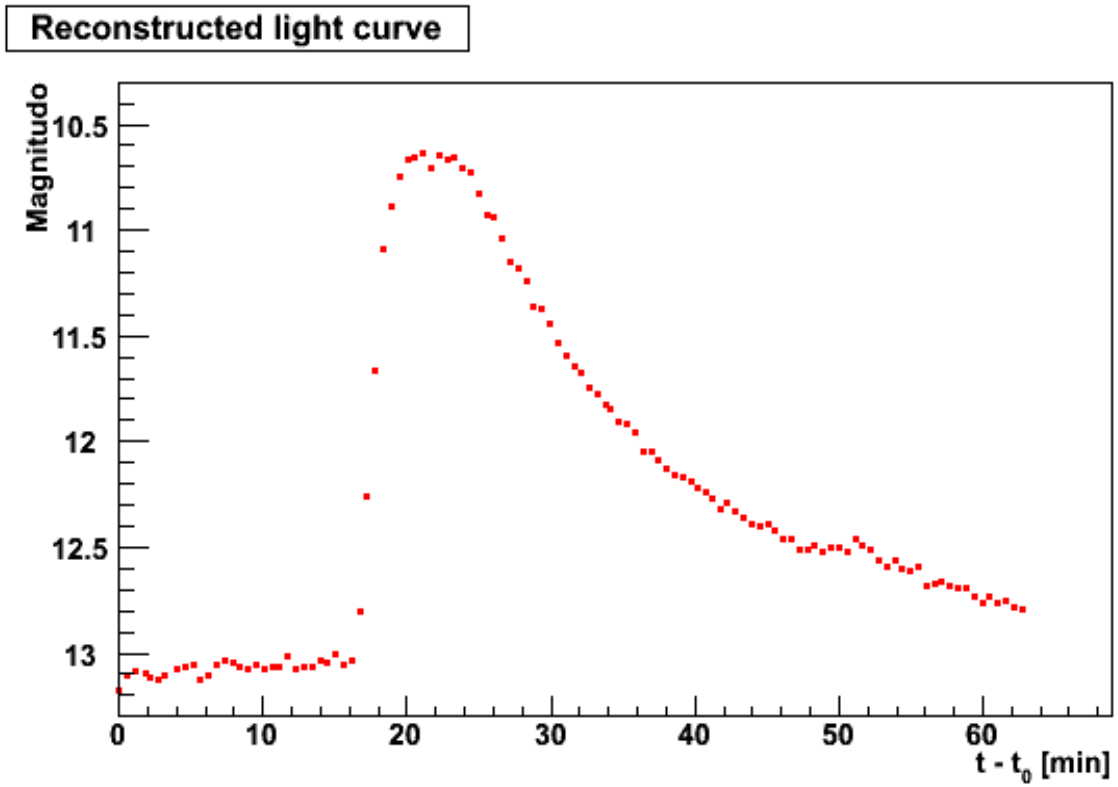
- Using Bootes-1 data taken on Nov. 25, 2011 (thanks to Martin Jelinek and Petr Kubanek)
- Processing procedure
  - Image reading
  - Image display
  - Pixel cluster finding
  - Object display
  - Object list storing



Working example

- Light curve obtained from the Luiza output

  Secondary outburst of a flare star RXJ0413.4-0139
  observed by Bootes-1 on Nov. 25, 2011
  (followup observation of the outburst observed by Pi of the Sky)



Documentation

- Documentation based on Doxygen
  - Web page and/or LaTeX document created from class header files, based on simple tags used in the comments included in the code
  - Easy to keep documentation up to date
  - Also code submitted by users can be easily added to the documentation

Luiza

v00-05

Welcome to the Luiza documentation page. This is the place where you can find some explanation and examples of how to run GLORIA analysis programs within Luiza.

Luiza framework was based on the Marlin package developed by Frank Gaede and Jan Engels from DESY for the ILC data analysis.
Please acknowledge their effort by refering to:
F. Gaede, Marlin and LCCD: Software tools for the ILC, Nucl.Instrum.Meth.A559:177-180,2006.

The idea (and large parts of code) taken from Marlin is that every computing task is implemented as a processor (module) that analyzes the data stored in a **gloria::GloriaDataContainer** structure and created additional output is also added to that collection. The advantage of such a modular approach is to keep things as simple as possible. Every single step of the full analysis chain that goes eg. from raw images to light curves can be processed step by step and the output of each step is still self consistent and can be fed in to the next step without any manipulation. Dedicated processors can also be included to read data from local file or external file server, access databases as well as to store analysis results. FIT image collections as well as tabular data collections can be stored. The framework allows to define the processors (and their order) that are executed at runtime in a simple XML steering file. Via the steering file you can also define named parameters (as a string, float, int - single and arrays) for every processor as well as for the global scope. By using the framework users don't have to write any code that deals with the IO they simply write processors with defined callbacks, i.e. init(), processData(), check(), end().

More information is available at the following pages:

- **How to install Luiza frmework**
- **How Luiza processes data**
- **How to prepare steering file**
- **Image analysis chain**
- **Writing your own processor**
- **GLORIA Sample Data**
- **Graphic User Interface for Luiza**

Last versions release info:

Version v00-05

- First version stored in Gloria SVN
- GUI for steering file editing added, based on QT4
- Processor for writing tables to text files added
- Image viewer processor based on CERN root libraries
- Exceptions allowing for job control from processor level implemented

Conclusions

- Efficient and flexible analysis framework for GLORIA developed based on HEP concept
- Basic structure and tools implemented, ready to work on GLORIA experiments
- First public version v00-05 available:
  - From GLORIA project SVN
    http://sourceforge.net/projects/gloriaproject/
    (go to misc → Tools → Analysis)
  - From **Luiza documentation web page**
    http://hep.fuw.edu.pl/u/zarnecki/gloria/luiza/doc/html/index.html
- First off-line experiment with Luiza in September