CIWS Software Specification Document (SSD)

Prepared by:	Name:	A. Bulgarelli et al.	Signature:		Date:	31 Mar 2014
Reviewed by:	Name:	A. Bulgarelli	Signature:		Date:	
Approved by:	Name:	M. Trifoglio	Signature:		Date:	

AUTHOR LIST

Andrea Bulgarelli, Andrea Zoli	INAF/IASF Bologna, Italy

DISTRIBUTION LIST

CIWS e-mail list	ciws@iasfbo.inaf.it
V 0.1	gtb@iasfbo.inaf.it
V 0.2-0.6	ciws@iasfbo.inaf.it
V 0.9	ciws@iasfbo.inaf.it

DOCUMENT HISTORY

Version	Date	Modification
V 0.1	10 June 2013	First draft.
V 0.2	19 June 2013	Comments from Massimo Trifoglio
V 0.3 and V 0.4	4 Jul 2013	Changes in QL section
V 0.9		First version on gdoc View, Canvas, Pad, Graph defintiions

TABLE OF CONTENTS

1. Introduction

The *Customizable Instrument Workstation Software (CIWS)* is a telescope - and instrument - independent software system for raw Level 0 (L0) and Level 1 (L1) data handling.

The system is aimed at providing a common and standard solution for the storage, processing, quick look and retrieval of the data generated by space-borne or ground-based telescopes and by the auxiliary data sources (e.g.: calibration facilities housekeeping) during the development, integration, verification, test and operation activities to be carried out on the telescope instrument.

1.1 Purpose of the document

The present document represents the output of the Software Requirement phase and of the Architectural Design phase, according to RD[1].

The purposes of this document are the following:

1. to serve as a tool to gather the software requirements that will drive the development of the software of this subsystem
2. it could be used as a guide to select and re-use existing software, to be identified at a later stage during the architecture definition of the subsystem
3. it should be used to identify the correct interfaces with external systems

It is important to indicate that the requirements in this document are just aimed at the software development process.

The SRD is an authoritative statement of what the software should do and it should be detailed enough to allow the implementation of the software without user involvement. This SRD is put under formal change control by the developer as soon as it is approved: new requirements may need to be added and old requirements may have to be modified or deleted.

The requirements are marked with the “SR” label.

The text not marked shall be considered as explanation text.

1.2 Definitions, acronyms and abbreviations

Name	Aliases	Labels	Documentation
data type			
data field			
data format			
graph			
view			

view group			
chain			
auxiliary data			In the "ground" context the auxiliary data are called "telemetry" and it is everything that is not an image. In the "space" context the auxiliary data is everything that is not produced by satellite
broadcast event			A pipeline event sent to the overall running processes.
DAS			The Data Acquisition System
data			Data are typically the results of measurements
data flow			The data flow is the flow of data (from input to output) managed by pipeline. A data flow is compound by different datatype.
data level	DL		It indicates the level of data after a transformation. A data level is DL0, DL1, DL2, etc
datatype	data type, DT		A data type define a type of data managed by IW system. The classification of DT depends by the instrument. e.g. EVT, CAL, AUX, HK
DL0			The acquired data level that contains the TM and TC generated by subsystem and P/L at Source Packet level. The scientific data, auxiliary data and housekeeping data in input to DPS. L0 are raw data. Everything in input to DPS
DL1			The data level that contains the data, calibrated and annotated raw, translated into FITS format for scientific analysis and elaboration. This is a processed data (by DPS).
DLx			A generic data level greater than 0 (DL1, DL2, etc). This is processed data.
DPS			Data Processing System

housekeeping data	HK		It refers to the data devoted to check the health and the status of the scientific instrument (voltage, temperature, etc.)
idle mode			An idle mode is a mode of the acquisition pipeline when the instrument is in pause or under configuration.
instrument workstation			The instrument workstation is a workstation that support the verification and monitoring activities to be carried out on the instrument during the development and operative phases of space-borne or ground-based telescopes. The system system is limited to data handling tasks, and does not include the instrument command generation and command handling tasks.
log			A record about actions or operations performed by the system.
measure	run, measurement		A set of data with a logic and scientific correlation. Each measure starts with a start event (typically a start command) and end with a stop event (typically a stop command). For example, a measure could be the data of a particular test or, during the calibration activities, the data of a particular source. Each measure is divided into two periods: 1) the *idle period,* the period before the start of the measure 2) the *run period,* the period of the measure The RUNID is an incremental number that identify a measure within a session.
measure log	measurement log, run log, measure_log		A log of the current measure written by the user containing the results of the acquired data
mode	period		A mode is the status of a measure. A measure is divided in run mode (when the acquired data are used for analysis) and idle mode (when the instrument is in pause or under configuration)
node			A node is a computer or machine where the data is acquired
off-line mode	deferred mode, off-line period,		In this mode the acquisition is performed in a different period with respect to the data flow of the instrument under test.

	deferred period		
on-line mode	live mode, on-line period		In this mode the acquisition is performed with the same rate and with the same data flow of the instrument under test.
pipeline event	event		Something that change the pipeline status
processor			A processor is a software component that convert the data from one format to another format (e.g. form L0 to L1)
quick-look software			A software that display the acquired data in a quick-look way
raw data	primary data		Raw data, or unprocessed data, is data collected from a source and has not been subjected to processing or any other manipulation. It is also referred as primary data.
run mode			A run mode is a mode of the acquisition pipeline when the data is between a start and a stop event. Usually the data acquired during the run mode is used for data analysis
RUNID			The RUNID is an incremental number that identify a measure within a session.
scientific data	experimental data		It refers to data acquired by instrument/payload
session log	session_log		A manual log written by users related to the main purpose of the session
start event			A start event is a pipeline event that change the status of the pipeline from idle mode to run mode.
stop event			A stop event is a pipeline event that change the status of the pipeline from run mode to idle mode.

1.3 References

RD [1] Guide to applying the ESA software engineering standards to small projects, ESA BSSC(96) Issue 1 May 1996.

RD [2] Tailoring of ECSS Software Engineering Standards for Ground Segments in ESA - Part A: Software Engineering, BSSC 2005(1) Issue 1.0 June 2005

RD [3] ECSS Packet Utilization Standard (PUS), ECSS-E-70-41A, 30 January 2003

RD [4] Gianotti F., Trifoglio M., DISCoS – a detector-independent software for the on-ground testing and calibration of scientific payloads using the ESA Packet Telemetry and Telecommands Standards, ADASS X Conference Proceedings, Boston 12-15 November 2000

RD [5] A.Bulgarelli, F.Gianotti, M.Trifoglio, *AGILE-ITE-SD-001, ProcessorLib Programmers Guide*, AGILE-ITE-SD-001, IASF Sez. di Bologna Report 349/02, September 2002

RD [6] A.Bulgarelli, F.Gianotti, M.Trifoglio, *AGILE-ITE-SD-003 ProcessorLib Detailed Design Report*, AGILE-ITE-SD-003, IASF Sez. di Bologna Report 351/02, September 2002

RD [7] Definition of the Flexible Image Transport System (FITS), March 29, 1999, NOST 100-2.0, NASA/Science Office of Standards and Technology

RD [8] A. Bulgarelli et al., *PacketLib 1.3.6 Programmer's guide*

RD[9] A. Bulgarelli et al., *PacketLib 1.3.2 Interface Control Document*

RD[10] Chen, P., et al., "The Entity-Relationship Model - Toward a Unified View of Data". ACM Transactions on Database Systems 1 (1): 9–36. doi:10.1145/320434.320440, 1976

RD[11] Grady Booch, James Rumbaugh, Ivar Jacobson: Unified Modeling Language User Guide, Addison-Wesley 1999

RD[12] Martin Fowler: Uml Distilled : Applying the Standard Object Modeling Language, Addison-Wesley 2003 (tratta UML 2.0)

RD[13] M. Trifoglio et al., CIWS URD

2. Model Description

This section should include a top-down description of the logical model. Diagrams, tables and explanatory text may be included.

The functionality at each level should be described, to enable the reader to 'walkthrough' the model level-by-level, function-by-function, flow- by-flow. A bare-bones description of a system in terms of data flow diagrams and low-level functional specifications needs supplementary commentary. Natural language is recommended.

1.4 View Model Design

The developer shall construct an implementation-independent model of what is needed by the user. This is called the **Logical Model** and it is used to produce the software requirement.

The Logical Model constructed in this document is based on 4+1 View Architectural model (see Figure 1) that will be used in the overall views and with more details for the development of the architecture.

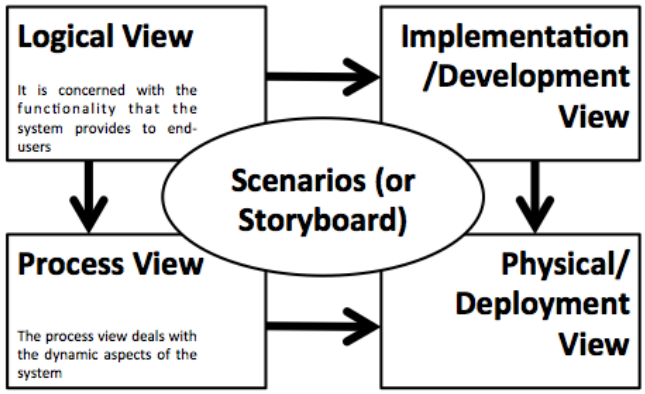

Figure 1 : 4+1 View Architecture Model

The **Logical View** describes the functional requirements of the software. It is an abstraction of the software that identifies the packages and the subsystems. This is a static point of view of the system. A fundamental input of this view is the Scenario View defined in [1].

The **Process View** describes the planned structure process of the system. The system is decomposed into a set of independent tasks/threads, processes and process groups. This view describes process interaction, communication, and synchronization.

The definition of the **Logical Model** is based on **Logical View + Process View**.

The **Implementation View** illustrates a system from a programmer's perspective and is concerned with software management.

In this document the definition of the Logical Model is based on block diagrams and object oriented design techniques described with UML diagrams.

A *function* is a defined objective or characteristic activity/action of a system that define the transformation to be performed on specified inputs to generate specified outputs. These transformations should be classified in

data reduction: a transformation to one type of data to another type

data analysis: producing final results, e.g. alerts, reports, plots, sky maps, etc.

The *functions* are connected by means of a *data flow*. The data flow should produce a *control flow* between functions (the output of a function controls the behaviour of another function).

In this section the functional decomposition of the system (Logical View) and the Process View in terms of interactions and communications (i.e. in terms of data flow) between functional block is reported. The Process View/Control Flow is not reported in this section.

Keywords:

Logical View/Functional Decomposition (static)

Logical View/Data Model (static)

Process View/Data Flow (dynamic)

Process View/Control Flow (dynamic)

1.5 Functional decomposition

1.5.1 Data Format, Data Type and Data Level

Keywords and acronyms:

DF = Data Format

DL = Data Level

DT = Data Type

A **Data Type** (DT) is a classification identifying a set of types of data (a **field**). Each field determines: (i) the possible values; (ii) the operations that can be done on values of that type; (iii) the meaning of the data. A field should be simple or compound (e.g. a vector). Data types are used within type systems, which offer various ways of defining, implementing and using them. To describe the meaning of the data, each field has a set of attributes. e.g. if a data type is 'housekeeping' with a set of temperatures for each sub-system (types), each temperature has a set of type attributes like min and max value, alarm thresholds, calibration curve, etc,

The **data format** (DF) is a physical realization of a data type (e.g., FITS file, database, etc). Basically it is a constraint placed upon the interpretation of data in a type system, describing representation, interpretation and structure of values or objects stored in computer memory.

The **data level** (DL) indicates the level of data after a transformation. The transformation should be a data format or data type transformation. The DPS input data level is in any case the DL0, regardless the structure and the format of the data itself.

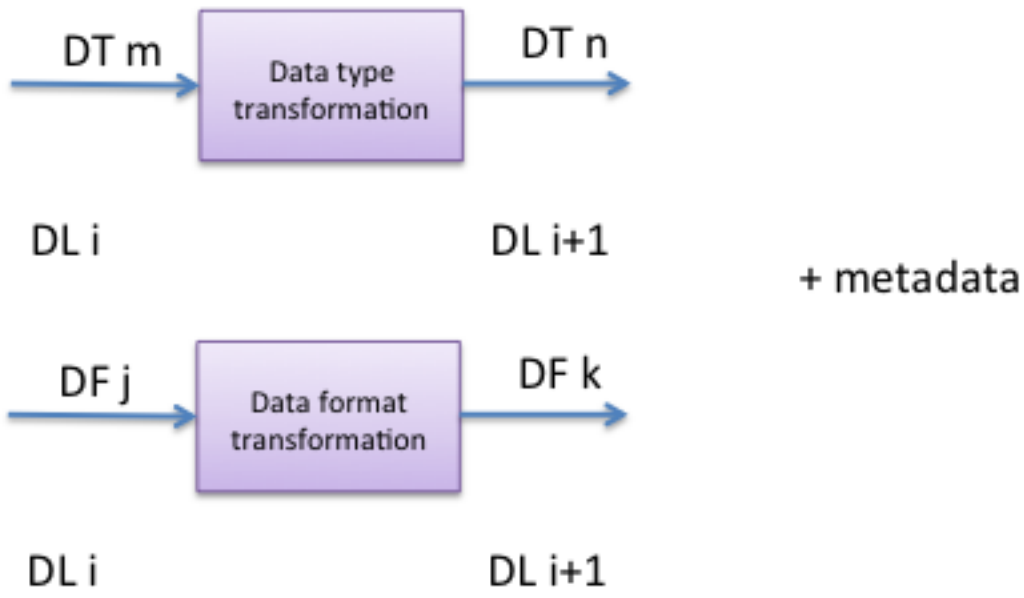

Figure 2: data transformation

The following are the *generic* data type that the system should acquire:

- scientific data
- housekeeping (and rate-meters)
- auxiliary data (everything that is not generated by the scientific instrument)

Data Definition Languages (DDLs), common for the entire CIWS framework describe the *data types* and the *data formats*.

1.5.2 Logical View/Functional Decomposition and Process View/Data Flow

To better clarify the software requirements we use the following **classification of functionalities (functions)** that should be performed by the IW and a short description of the main capabilities:

1) **Data Processing System (DPS)**

   A. **Data Acquisition (DAQ)**:

a. the acquisition of the TM and the TC streams coming from instrument (the DL0 data format);
b. the acquisition of files coming from instrument (the DL0 data format)

B. **Data Transformation (DTR)**: the conversion of the DL0 acquired data into DL1 data format (data type and/or data format transformation);

C. **Quick Look (QL)**: the display of the data coming from the instrument both in L0 and L1 format. In particular, related to the L1 display the *Quick Look (QL)* software is able to perform the following functions:
- to process and display the **scientific data** coming from the instrument *for engineering verification purposes*. These information shall be organized in *views*. In some cases is should be present some additional scientific elaborations for the display of scientific data as support of the engineering verifications;
- to perform the health assessment of the instrument by means of the **house-keeping data** display, in particular with the display also of the **ratemeters**;
- to display sky maps (a scientific data) with overlapped reference catalogues (if sky maps are provided as DL0 or DL1);

The QL software should perform also data analysis starting from L1 data.

The **data display and analysis** should be done in a **simplified** or in a **full** mode. In both cases the data analysis should be **automated** or **interactive**.

2) **Data Archiving System (DAS)**:
   a. archiving of the DL0 and DL1 data
   b. queries (on metadata) and data retrieval (of data) as result of queries.

3) **Reference Catalogues**: a set of celestial objects catalogues provided to QL for a manual check of scientific data

4) **Logging System**: to record events occurred during or after the data acquisition to understand the activities performed and the results obtained with the measure. The logs are edited manually by the user and are also automated.

These functionalities shall be provided mainly in **on-line mode** (during the data acquisition) and in **near real-time**, and in **off-line mode**.
The term “near real-time” is extensively used in most contexts. This term do not indicate a real-time system, it is used to specify “as soon as possible”. In some cases on-line mode and near-real-time mode should be used as synonymous.

In the following, the term **instrument** refers to any kind of source of data of any data type and data format. In any case, the instrument generates the DL0 data. The context is described in RD[13].

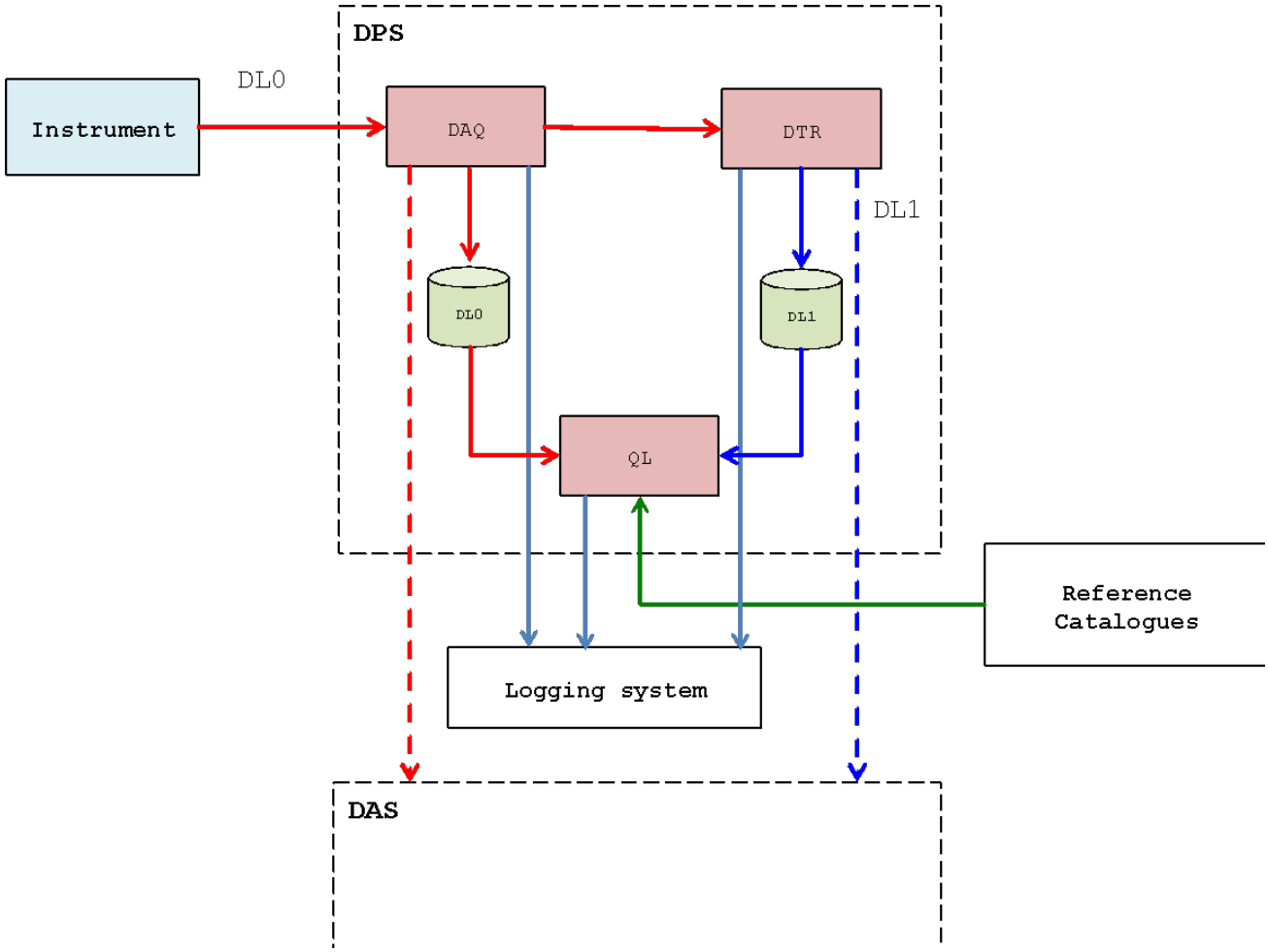

Figure 3: The Logical View of the IW in **on-line data processing mode**. This diagram shows the relation between functional units.

Figure 2 shows the logical view of the system in on-line mode. The DL0 data is acquired by DAQ. The DAQ send the DL0 data to

A. a local DL0 data archive
B. the DTR
C. the DAS (if present)

The DTR stores the data into a DL1 local archive and send the data to DAS. Through a sync mechanism (not shown) the data are read from DL1 local archive and displayed with the QL. As alternative data flow (not shown) the DTR should send directly the data to QL.

The QL should use the reference catalogues as input.

DAQ, DTR and QL store logging information into the logging system. The DAQ and DTR logging information are generated automatically; the IW user generates the QL logging information.

The logging system should be stored by DAS.

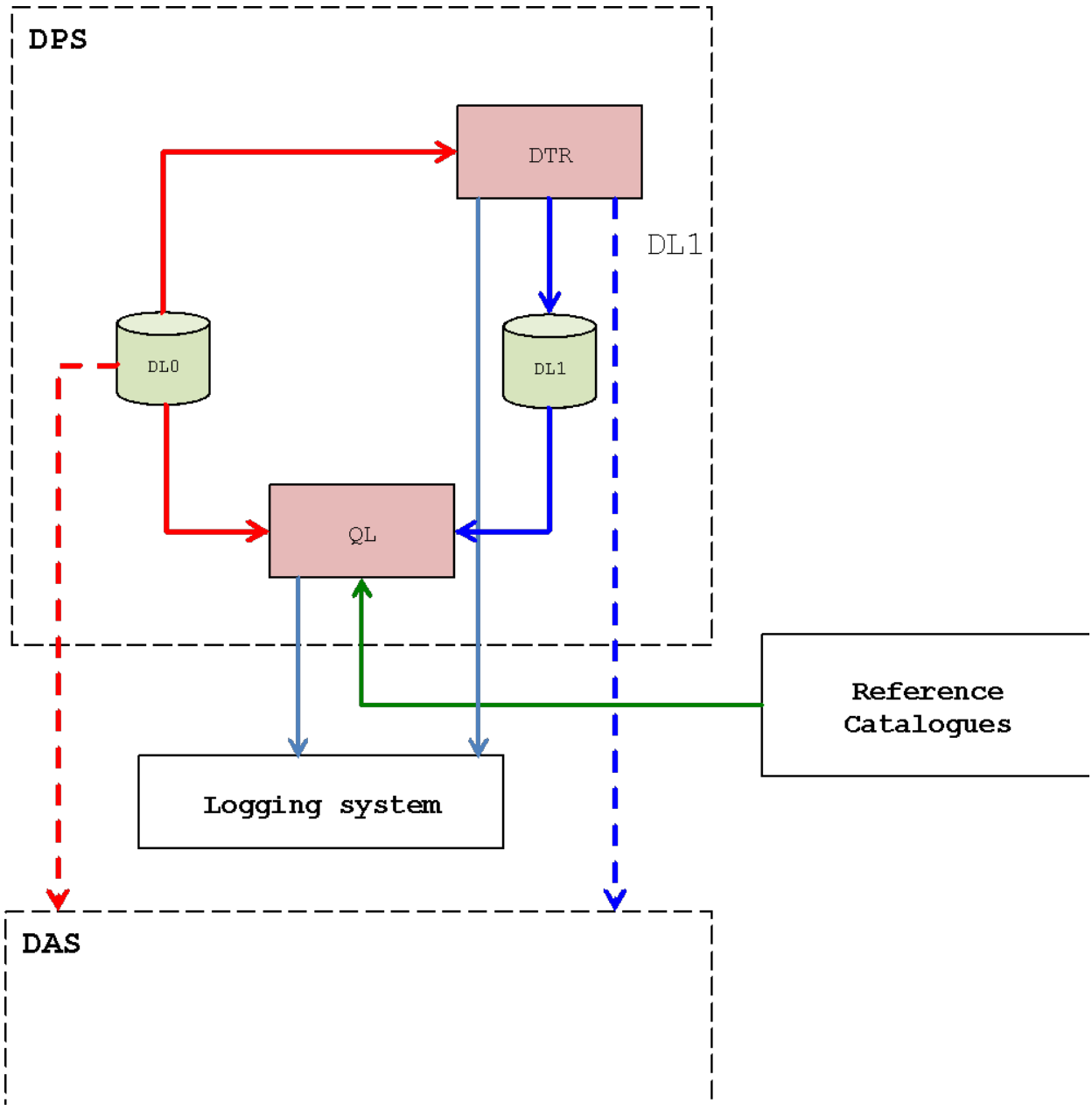

Figure 4: The Logical View of the IW in **off-line/reprocessing from DL0 local archive**. This diagram shows the relation between functional units.

The Figure shows the logical view of the IW in off-line mode with data reprocessing. The data flow starts from DL0 local archive. The reprocessed DL0 data should be sent also to DAS. The remaining data flow is the same of the on-line mode.

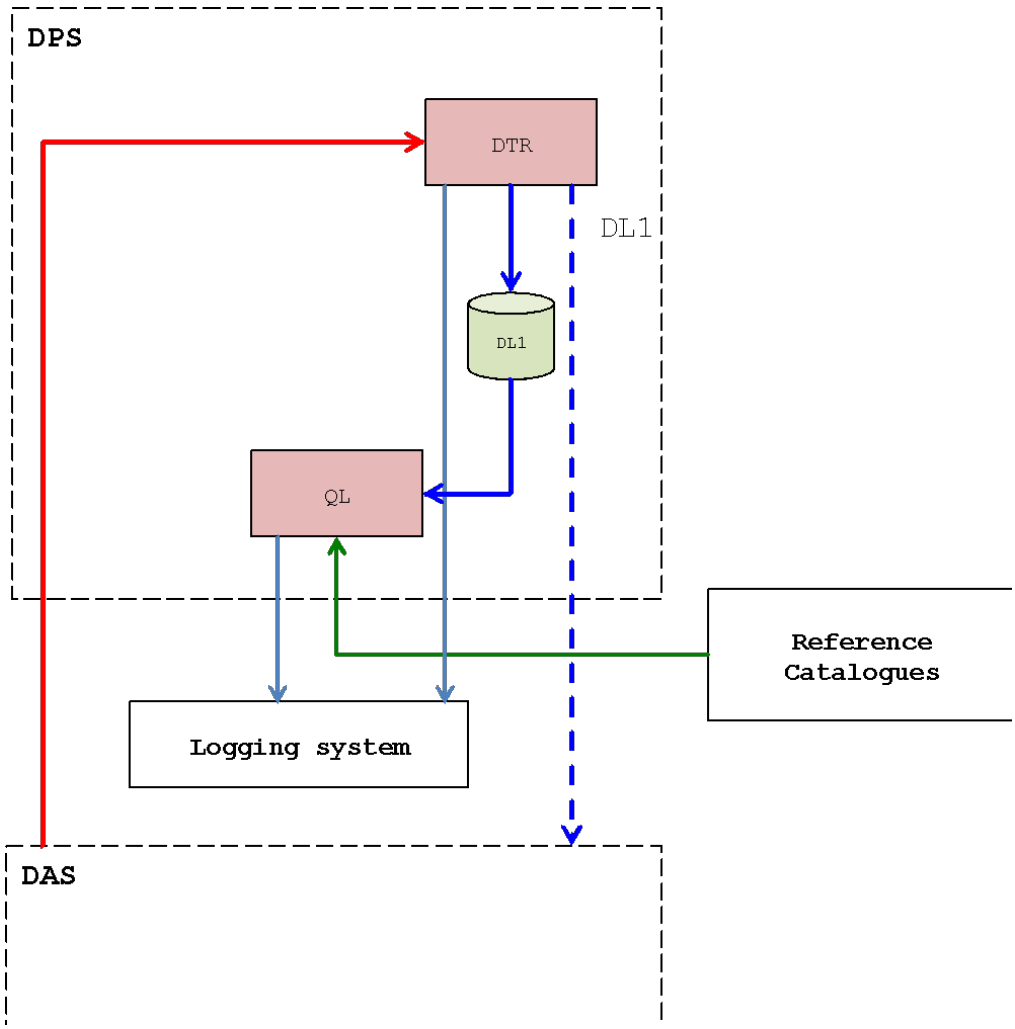

Figure 5: The Logical View of the IW in **off-line/reprocessing from DAS/DL0 archive** for data reprocessing and QL.

Figure shows the Logical View of the IW in off-line mode where the DL0 are reprocessed. The DL0 are retrieved from DAS. The remaining data flow is the same of the on-line mode.

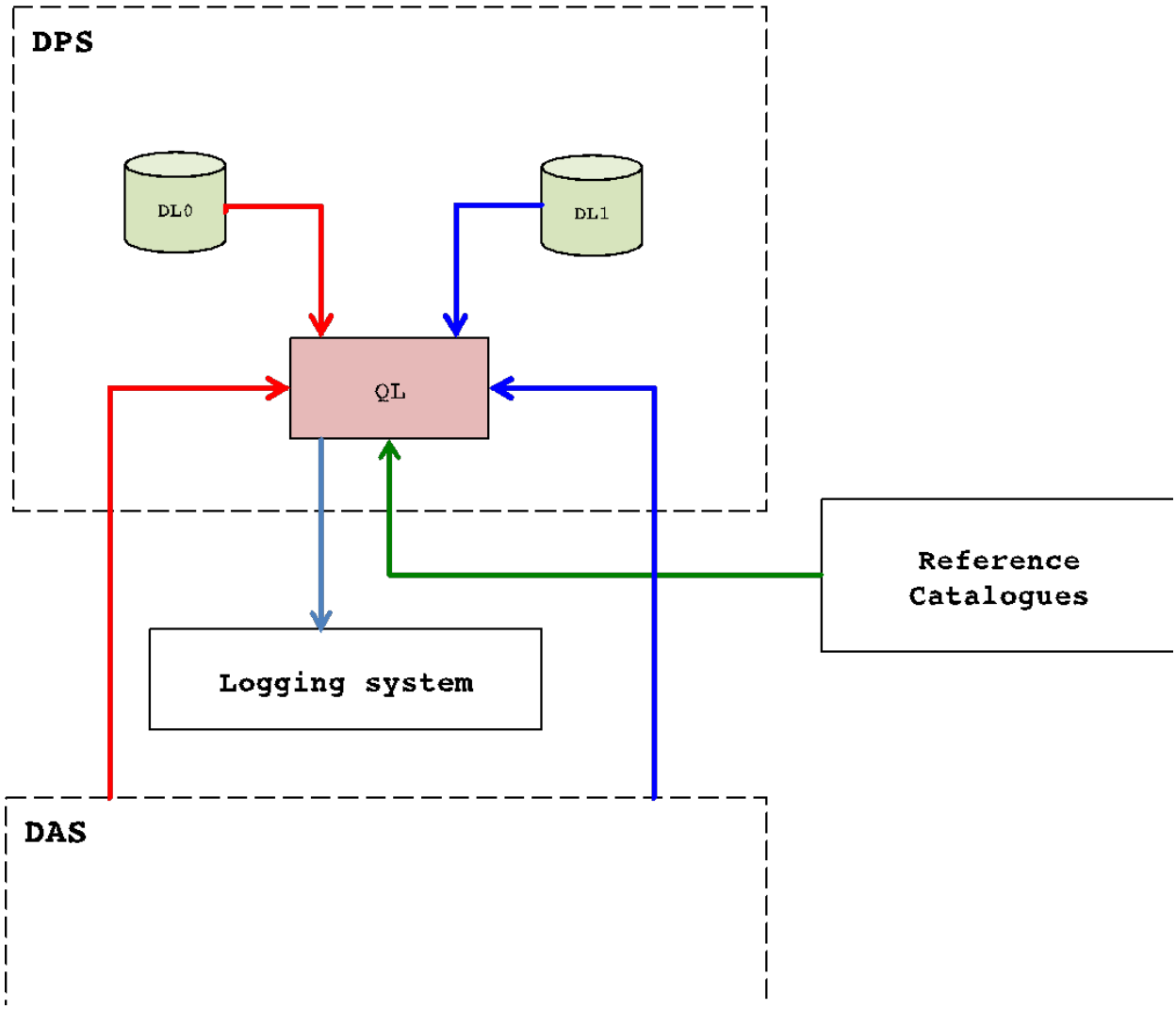

Figure 6: The Logical View of the IW in off-line mode for QL.

The QL shall work without the DPS functional block, reading directly the local archives (DL0 and/or DL1) and the DAS archive system performing queries on data and data retrieval for off-line analysis.

1.5.3 Logical View/Data Model

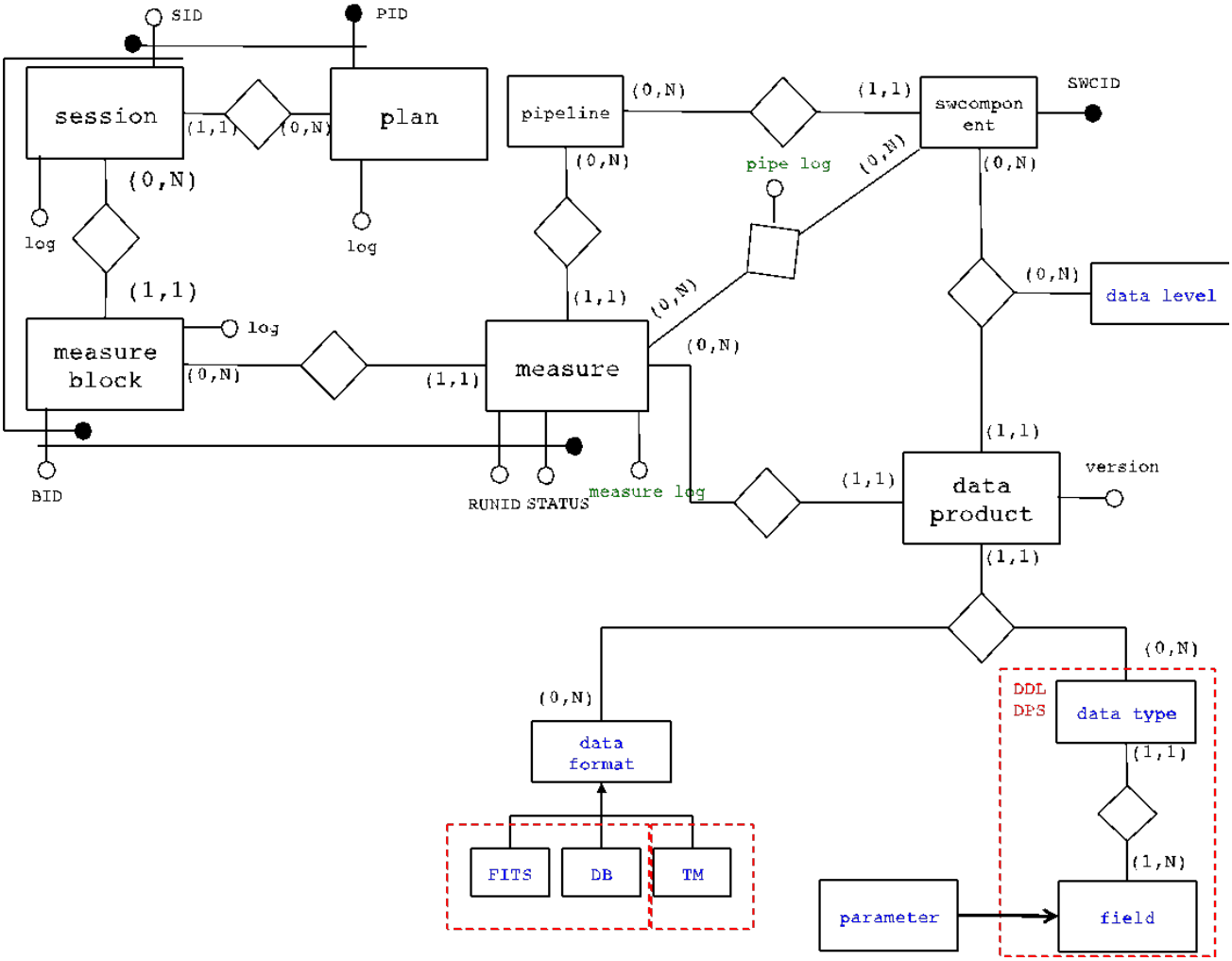

Figure 7: The Logical View/Data Model using E/R notation (RD[10]).

Each **measure** shall be identified by (see CIWS URD for details):

- plan ID (PID)
- session ID (SID)
- measure block ID (BID)
- RUNID
- status (idle, run, etc)

A measure has a set of **data products.** The version of the data product is used to identify the products obtained by different data reprocessing.

A **pipeline** is compound by **software components** (or module). Each software components has input, output and a well defined functionality. A software component should generate an automated log during the data acquisition. A software components store the data (**data product**) in a well defined **data format** (e.g. FITS file, into a DBMS, etc) of a well-defined **data type**. The same **data type** should have different data format. E.g. if a data type is 'housekeeping', the same data type should be used (and stored) as DL0 binary format, DL1 FITS format and into a DBMS.

The **data level** depends by the pipeline configuration.

The **data types and the data formats** are described with Data Definition Languages.

In addition, each Data **Type** has a set of fields that determines the possible values, the operations that can be done on values of that type, the meaning of the data. To describe the meaning of the data, each type has a set of attributesA mapping mechanism between high level data type and data format is provided by the system.

2. Specific Requirements

This sections lists the specific requirements, with attributes.

This Section describes the CIWS software requirements that are categorized as recommended in [RD1].

Each requirement will be identified by a unique identification code obtained by the union of the CIWS-SR and an incremental number.

A short descriptive qualifier is also provided as the title of each requirement.

Three categories for the standard practices are used:

. mandatory practices: sentences containing the word '**shall'**: this must be traced and verified
. recommended practices: sentences containing the word '**should'**: no compliance reason shall be asserted. Verification could be omitted
. guideline practices: sentences containing the word '**may'**: no compliance reason could be asserted

Each software requirement is described by:

- an identifier
- a short title
- a description in natural language
- a motivation (Motivated by)

For functional requirements, the description also includes

- the input
- the output
- the activation conditions

This is a template of functional requirements:

CIWS-SRx.yyyy *"Data acquisition"*

The system shall be able to …

Motivated by:

Activation conditions:

Input: the DL0 data level

Output: the DL1 data level

2.1 Functional requirements

A function is a defined objective or characteristic action of a system or component and a functional requirement specifies a function that a system or system component must be able to perform. Functional requirements are matched one-to-one to nodes of the logical model. A functional requirement should:

- *define what a process must do, not how to implement it*
- *define the transformation to be performed on specified inputs to generate specified outputs*
- *have performance requirements attached*
- *be stated rigorously*

*A function is described as a set of **inputs**, the **behaviour**, and **outputs**. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Functional requirements are supported by non-functional requirements, which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "**system shall do <requirement>**".*

Functional requirements identify specific pieces of functionality that must be built into an overall software product. In short, they indicate what a system should be able to do. These specifications can be high-level (i.e., contain general information) or low-level (i.e., contain lots of detail). For project managers and IT staff, the important thing is that, functional requirements provide a way to track progress toward the completion of a project. When written correctly, they clearly illustrate the work that's left to do versus the work that's already been done while helping to control project scope.

2.1.1 General Requirements of the IW

CIWS-SR-FR-0010 *"IW main functionalities"*

The IW system has the following purposes:

(i) data acquisition of DL0 data level,
(ii) data transformation of DL0 in to DL1 data level
(iii) data archiving
(iv) quick look of acquired data (both DL0 and DL1 data levels)
(v) user logging system

CIWS-SR-FR-0020 *"IW Data Level"*

The **data level** (DL) indicates the level of data after a transformation. The transformation should be a data format or data type transformation. The DPS input data level is in any case the DL0, regardless the structure and the format of the data itself.

The IW system works with two data levels:

a) DL0: the input of the system. The data type
b) DL1: a conversion of DL0 data level. The data level that contains the DL0 data translated into a format readable by humans for scientific analysis and elaboration.

CIWS-SR-FR-0030 *"IW Data Type"*

A **Data Type** (DT) is a classification identifying a set of types of data (a **field**). Each field determines: (i) the possible values; (ii) the operations that can be done on values of that type; (iii) the meaning of

the data. A field should be simple or compound (e.g. a vector). Data types are used within type systems, which offer various ways of defining, implementing and using them. To describe the meaning of the data, each field has a set of attributes.

The system shall work with the following data

1) scientific data coming from instrument/telescope (SCI): scientific data and related calibration data
2) housekeeping data (HK) and rate-meters (RT): not SCI data that come from instrument/telescope
3) auxiliary data (AUX): data that come from auxiliary items

Each data is mapped to a data type.

A DDL shall describe the data type managed by the IW.

A data type is a set of **fields**. A field should have at least the following attributes:

- name
- measure unit
- basic data type (integer, float, boolean, etc.)

CIWS-SR-FR-0040 *"Monitoring Parameters or Parameter"*

A field of a data type should be a **monitoring parameter** (or **parameter**). A monitoring parameter is a field with should be monitored and could generate alarms.

A monitoring parameter should have the following attributes:

- calibration curve
- min range for warning
- max range for warning
- min range for alarm
- max range for alarm
- lookup table
- see **CIWS-SR-FR-5370**

CIWS-SR-FR-0050 *"IW Data Format"*

The **data format** (DF) is a physical realization of a data type (e.g., FITS file, database, etc). Basically it is a constraint placed upon the interpretation of data in a type system, describing representation, interpretation and structure of values or objects stored in computer memory.

The DL0 data should be telemetry packets RD[3] or files in any format or data stored into a DBMS.

The DL1 data should be files in any format or data stored into a DBMS.

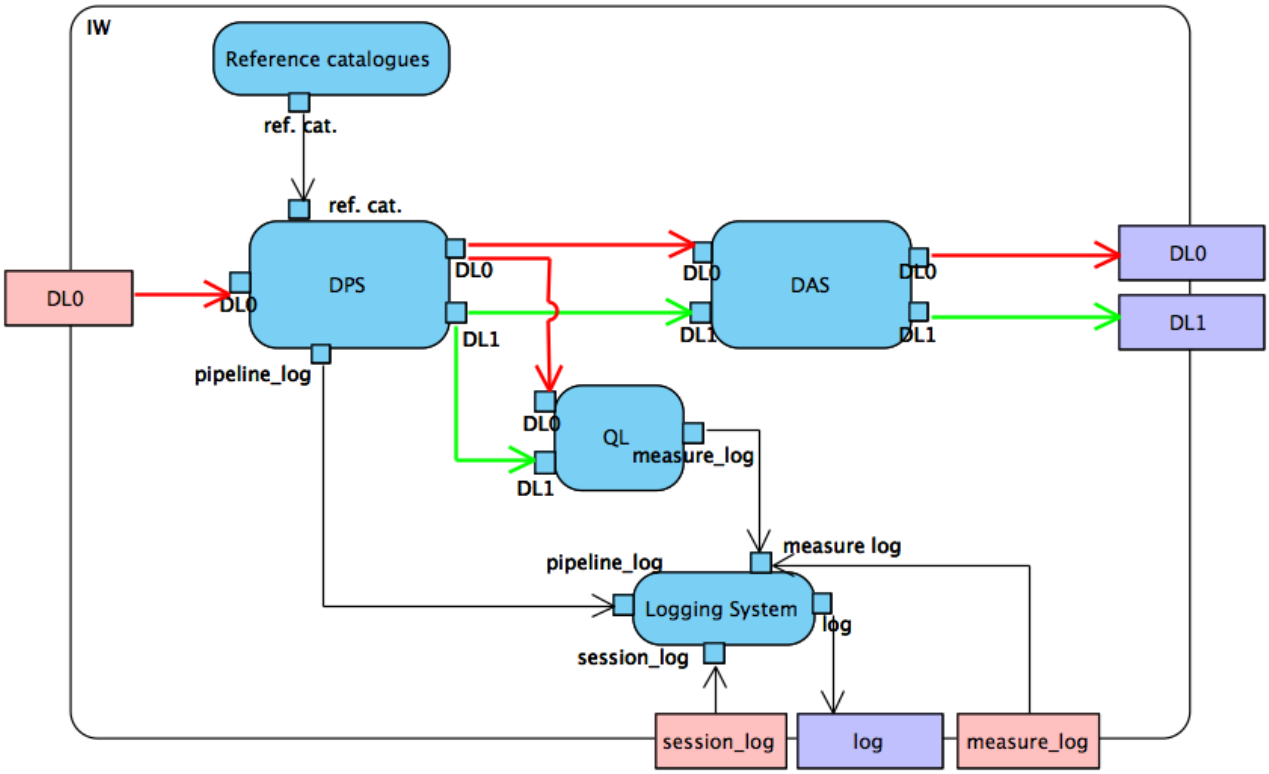

Figure 8: IW Activity Diagram with "Data Level" data flow: on-line data processing mode

CIWS-SR-FR-0060 *"IW main functional blocks"*

With reference to **Figure 8** the main functional blocks are the following**:**

1. **Data Processing System (DPS)**: DL0 data acquisition, DL1 data conversion and processing
2. **Quick-Look**: quick-look, measure log through quick-look
3. **Data Archiving System (DAS)**: DL0 and DL1 data storage and data retrievale
4. **Reference Catalogues**: query reference catalogues for QL purpose
5. **Logging System**: read/write/query logs of pipeline/session/measure

Input: DL0, session log, measure log

Output: DL0, DL1, logs

CIWS-SR-FR-0070 "*IW Data Processing mode*"

The IW system shall work in two *data processing modes*:

1) **on-line mode**: during the data acquisition

2) **off-line mode**: after the data acquisition

Not all the functional blocks work with these two data processing modes.

CIWS-SR-FR-0080 *"IW Data Processing mode: on-line mode"*

(see **Figure 8**) The DPS organize session of measures. Each measure is defined by a start and stop event. The DPS shall receive DL0 data and during the data acquisition/during the measure the data are converted into DL1 and stored locally. During the data acquisition a quick-look of the data shall be performed and result stored in the measure log.

At the end of each measure the DL0 and DL1 data are stored by the DAS.

2.1.2 DPS

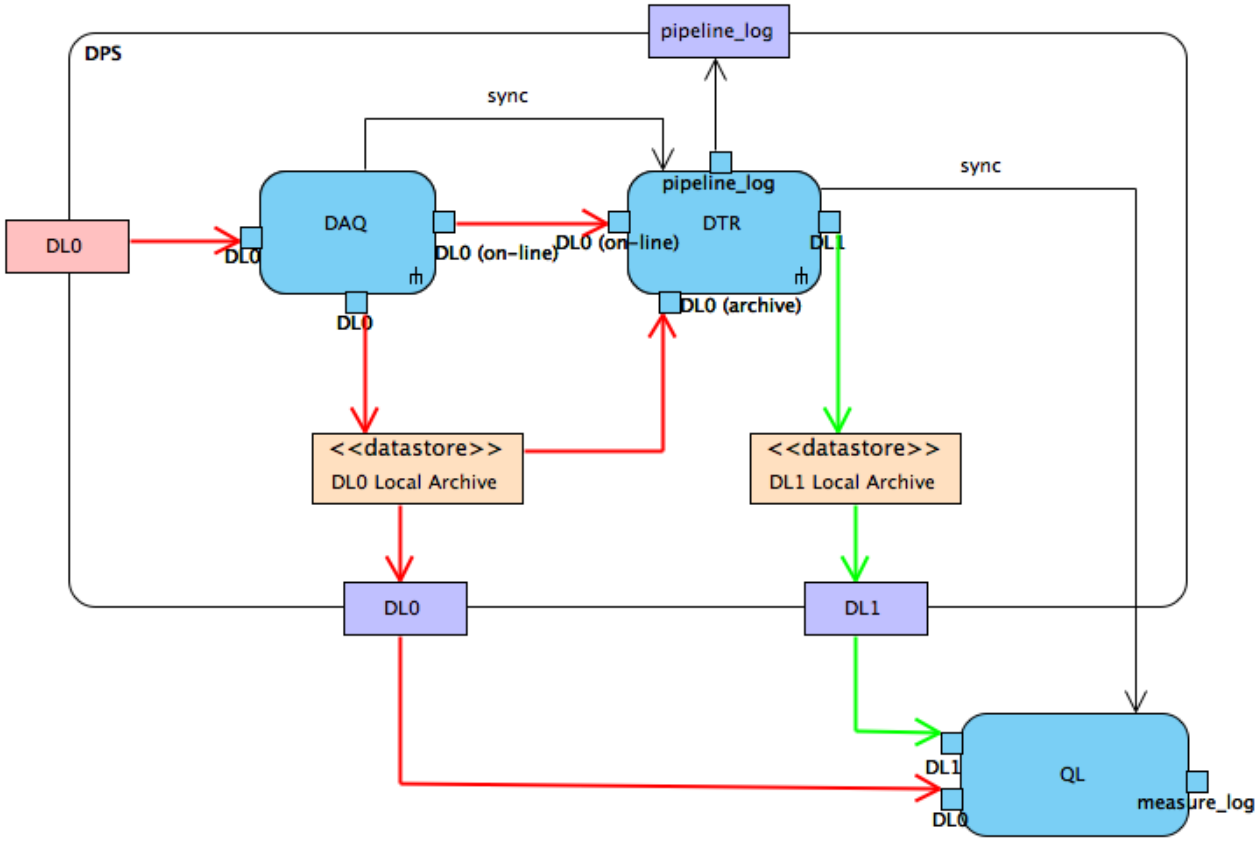

Figure 9: IW Activity Diagram of the DPS and related data flow.

CIWS-SR-FR-0090 "*DPS main functional blocks*"

With reference to **Figure 9** the main functional blocks are:

A. Data Acquisition (DAQ): DL0 data acquisition and local archiving

B. Data Transformation (DTR): data conversion from DL0 to DL1 and local archiving

C. Quick Look (QL)

D. DL0 Local Archive

E. DL1 Local Archive

CIWS-SR-FR-0100 "*DPS Working mode*"

The DPS sub-system shall acquire, process data **event-by-event**. If a block of event is provided the DPS sub-system may be able to acquire and process the **block of events-by block of events**.

The definition of the "event" depends by the instrument. E.g. in a CCD camera an "event" is a single acquired image. In a gamma-ray Silicon Tracker a single event is the pair converted and sampled photon. Basically, the definition o the event depend by the acquisition mode or trigger system of an instrument.

CIWS-SR-FR-0110 "*DPS Data Processing mode*"

The DPS system shall work in two modalities:

1) **on-line mode**: during the data acquisition

2) **off-line mode**: after the data acquisition.

2.1) off-line/reprocessing from DL0 Local Archive (**Figure 10**)

2.2) off-line/reprocessing from DAS archive (**Figure 11**)

2.3) off-line/*data retrieval mode (for QL)*

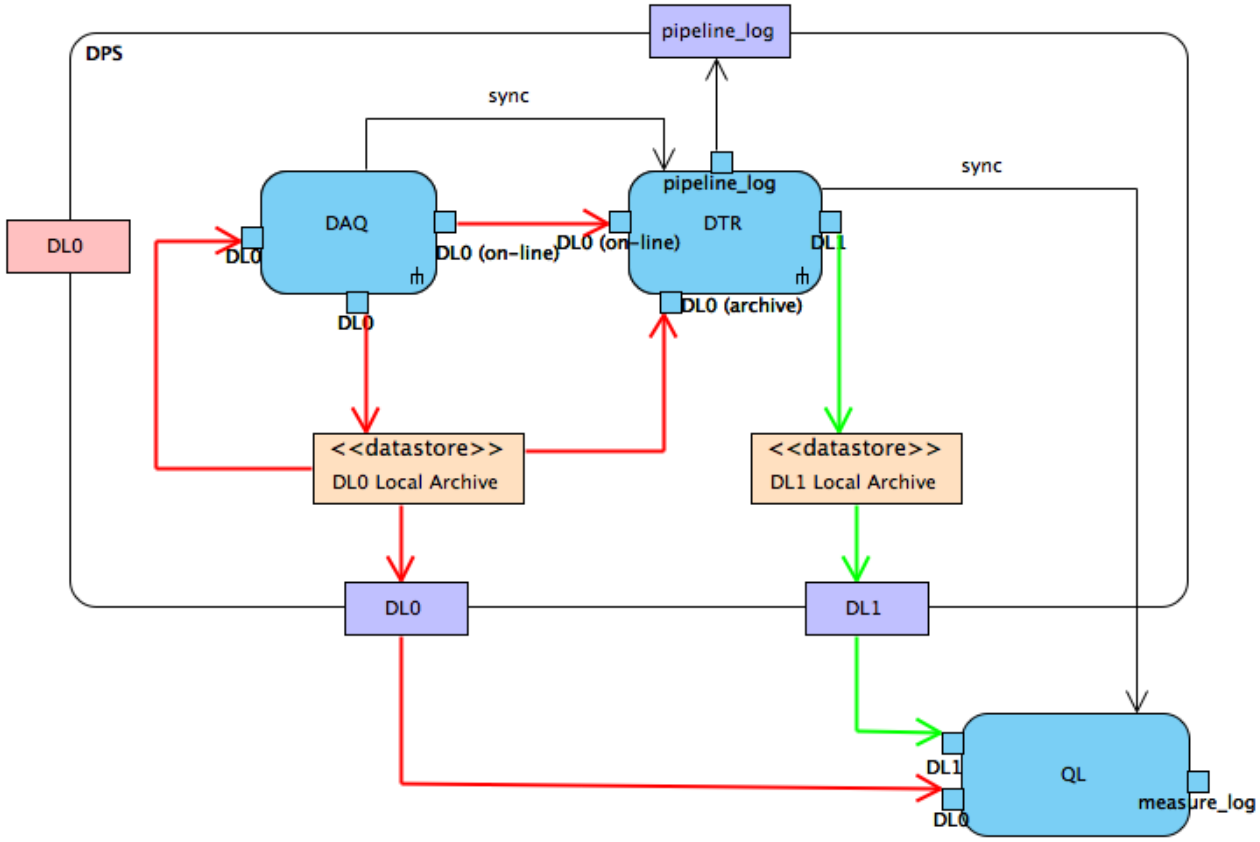

Figure 10: IW Activity Diagram of the DPS and related data flow: off-line/reprocessing from DL0 Local Archive.

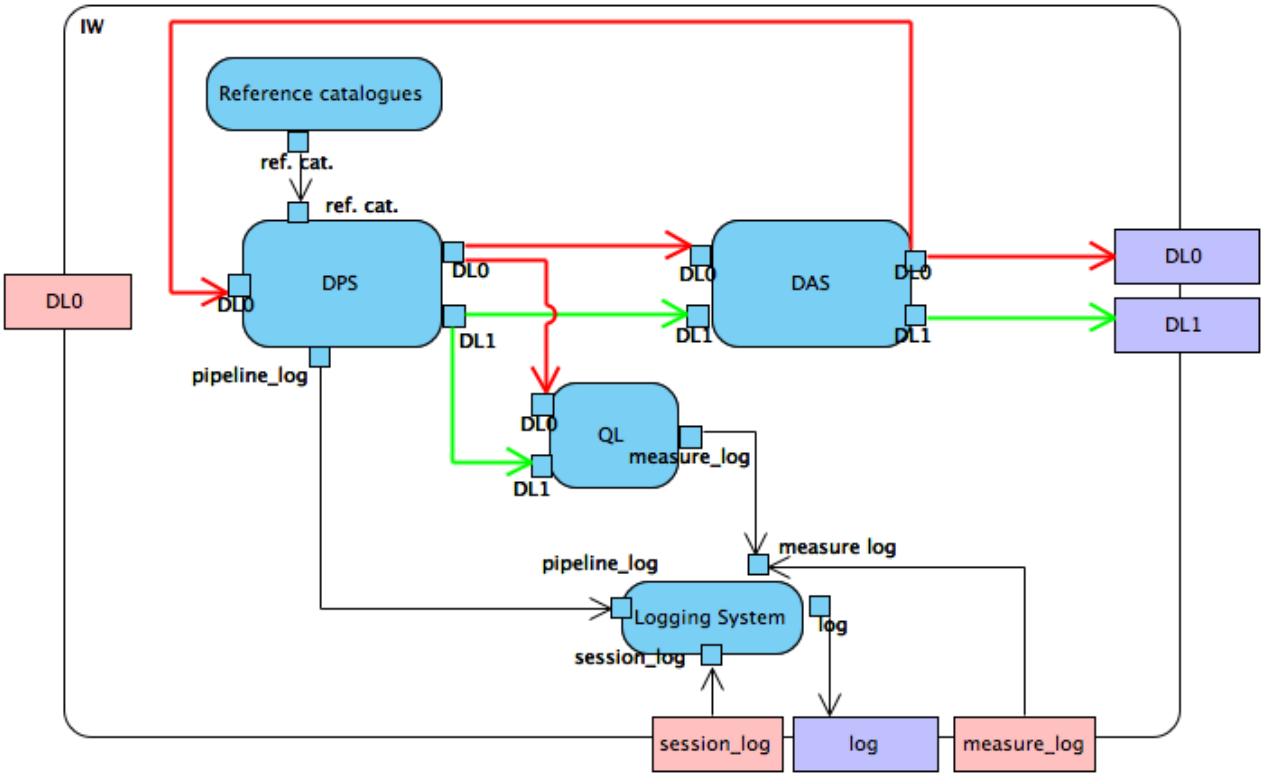

Figure 11: IW Activity Diagram of the DPS and related data flow: off-line/reprocessing from DAS archive.

2.1.2.1 DAQ

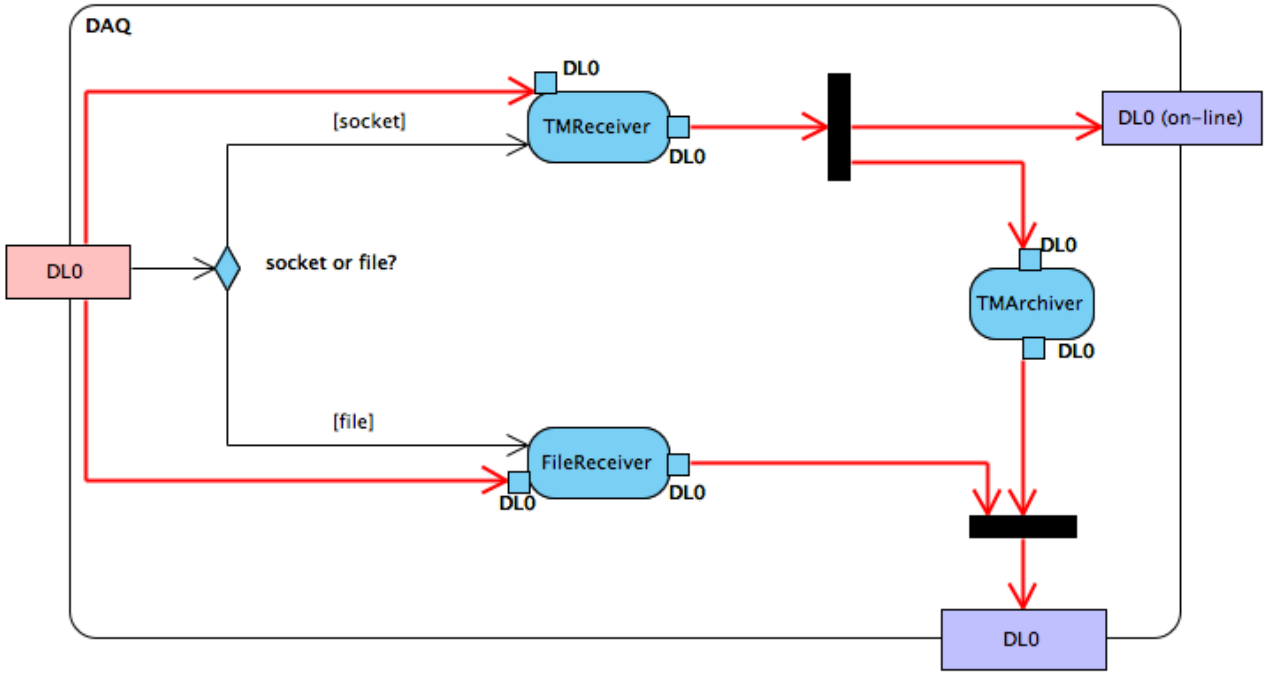

Figure 12: IW Activity Diagram of the DAQ and related data flow.

CIWS-SR-FR-0120 *"DAQ and local archiving"*

The system shall be able to acquire the DL0 data format and store the data in a local archive.

CIWS-SR-FR-0130 *"DAQ format and mechanism"*

The data acquired by the DAQ subsystem shall be telemetry (TM) and telecommand (TC) source packets, compliant with the Telemetry & Telecommand ESA standards or any type of files sent to IW via different communication mechanism (e.g. ftp or sftp).
Also data stream not ESA compliant may be managed by the system.
The source packet should be acquired via socket or file.

CIWS-SR-FR-0140 *"DAQ management"*

The DAC subsystem shall work with the logic of sessions and measure and manage them.

A *session* is a set of measures.

A *measure* is a set of data with a logic or scientific correlation. Each measure starts with a start event and end with a stop event.

Each measure is divided into two periods:

1) the *idle period*, the period before the start of the measure

2) the *run period*, the period of the measure

CIWS-SR-FR-0150 *"DAQ Session identification*"

See *Logical View/Data Model* for the identification of the session. The session identification is used by the overall IW system.

CIWS-SR-FR-0160 *"DAQ Measure identification*"

Each measure of a session should be identified by a unique incremental number, called RUNID. The boundary of a measure is identified by a start event and by a stop event.

CIWS-SR-FR-0170 *"DAQ Notification system"*

The DAQ should notify the DTR system when new DL0 data are present and coordinate the data processing of the DTR when new session or measure is present.

CIWS-SR-FR-0180 "*DAQ main functional blocks"*

With reference to **Figure 12** the main functional blocks are:

1. TMReceiver
2. FileReceiver
3. TMArchiver

2.1.2.2 DTR

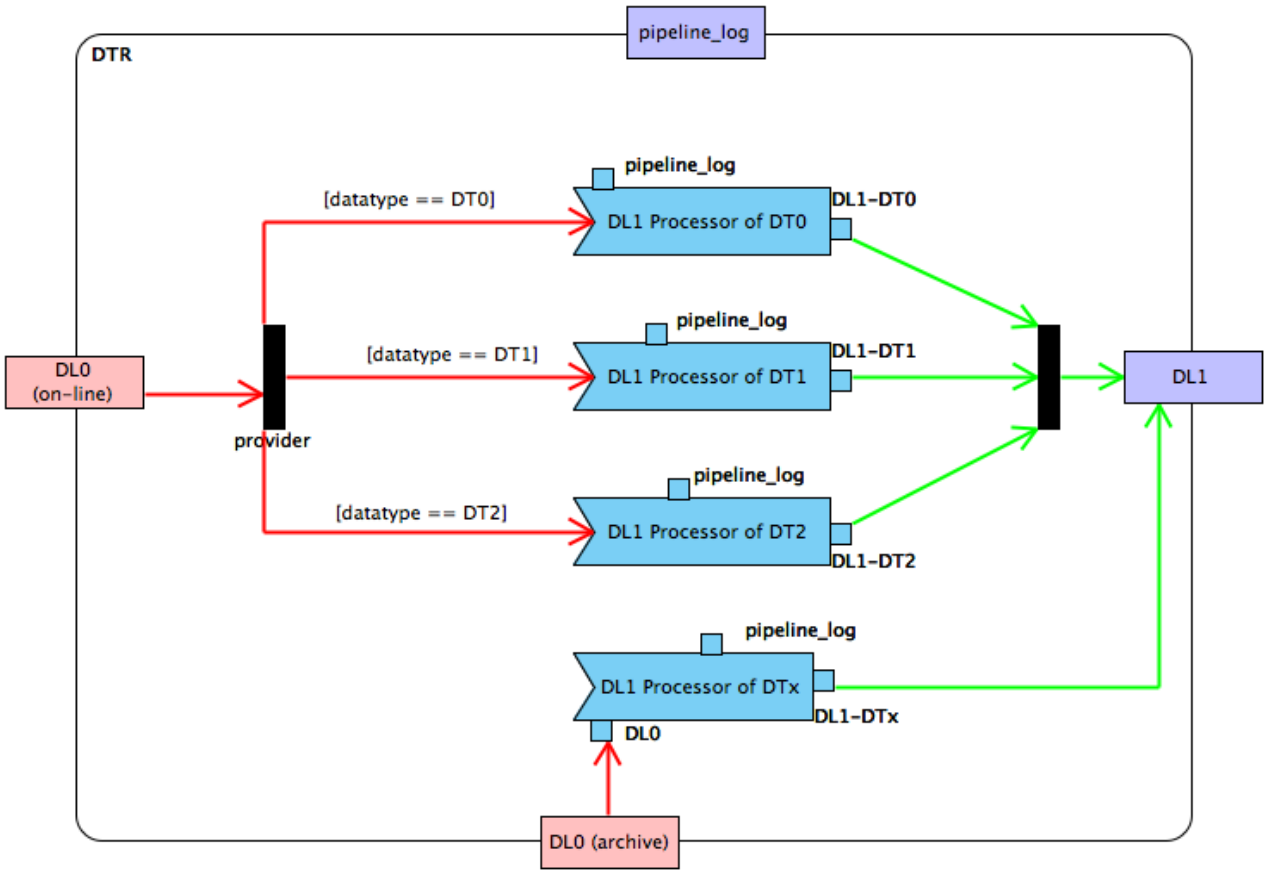

Figure 13: IW Activity Diagram of the DTR and related data flow. The data flow from pipeline_log activity nodes is not shown.

CIWS-SR-FR-0190 *"DTR Data Processing"*

The DL0 data shall be converted into DL1 data format. The conversion should be a data type or a data level transformation.

CIWS-SR-FR-0200 *"DTR Functional unit"*

A functional unit called **processor** shall perform the conversion**.**

CIWS-SR-FR-0210 "*DTR Data Processing Chains"*

Each processor is able to realize a single **Data Processing Chain**

2.1.2.3 Quick-Look (QL)

CIWS-SR-FR-5000 "*QL main functionalities: display and monitor*"

It shall be possible to **display** all the *data type* managed by the IW regardless the *data level* and the *data format*.

It shall be possible to **monitor** all the parameters.

CIWS-SR-FR-5010 "*QL Data Processing mode*"

The QL system shall work in two modalities:

1) **on-line mode**: this means that the system shall display the data acquired from the instrument during a measure. *The software shall be synchronized with the DTR*.

2) **off-line mode**: it is possible to load a stored data (a data product) to display the acquired data (**Figure 14**).

2.3.1) data retrieval mode (for QL) from Local/DL0 Archive

2.3.2) data retrieval mode (for QL) from Local/DL1 Archive

2.3.3) data retrieval mode (for QL) from DAS/DL0 Archive

2.3.4) data retrieval mode (for QL) from DAS/DL1 Archive

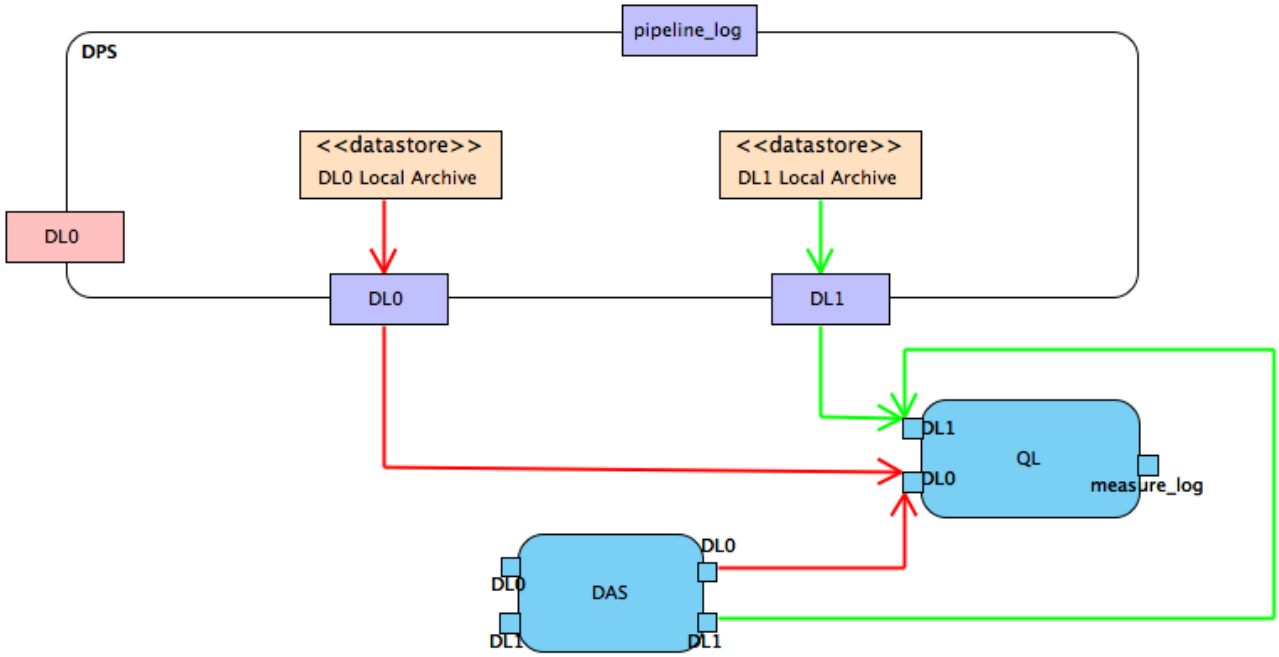

Figure 14: IW Activity Diagram of the QL off-line mode.

CIWS-SR-FR-5020 "*QL Working mode*"

The QL sub-system shall display data **event-by-event**.

CIWS-SR-FR-5050 "*QL Data Analysis mode*"

The QL system shall work in two modalities:

1) **full mode**: all the data acquired are displayed.

2) **sample mode:** a sample of the data acquired is displayed. The user should select the rate of the sampling (one event each N).

CIWS-SR-FR-5060 *"QL Refresh mode"*

The views of the QL are updated when a refresh event occur. To perform this task a **manual refresh mode** (performed by the user in both on-line and off-line data processing modes) and two **automatic refresh modes** are foreseen:

A. **push mode**: the view are updated automatically when new data (new events) are present. A synchronization mechanism should be provided by DTR. This mode works only in on-line data processing mode.
B. **pull mode** (or polling): the QL check the presence of new data. This refresh mode works in both on-line and off-line data processing mode.

CIWS-SR-FR-5070 *"QL Manual Refresh mode"*

It is performed by the user in both on-line and off-line data processing modes, with two sub-mode:

1.1.1) **event-by-event mode**, in which a single event is read from a single DTR chain. During this mode the other chains are ignored.

1.1.2) **step-by-step mode**, in which a block of events are read from all the DTR chain, starting from the last reading point to the next reading point of each DTR chain. The next reading point is established by

1. the number of events to read for each chain. The event-by-event mode is a particular case (where number of events is equal to 1).
2. Periodically every *n* seconds

1.1.3) **free-run mode** all the data product is read and displayed from the start to the end of the measure.

CIWS-SR-FR-5080 *"QL Automatic Pull Refresh mode"*

The **automatic pull refresh mode**, is time-driven and activated at regular interval (selectable for each chain). It starts from the last reading point to the last event acquired for each DTR chain.

CIWS-SR-FR-5100 "*QL Display data starting point*"

The software shall be able to refresh the data coming from the DAQ sub-system with the option of restarting the visualization from the current data. A full reset of the data acquired before the starting point is performed.

CIWS-SR-FR-5110 "*QL manual reset*"

The software shall be able to reset all the views manually.

CIWS-SR-FR-5150 *"Relationship between data type, data format, data view"*

See Figure **Figure 15**

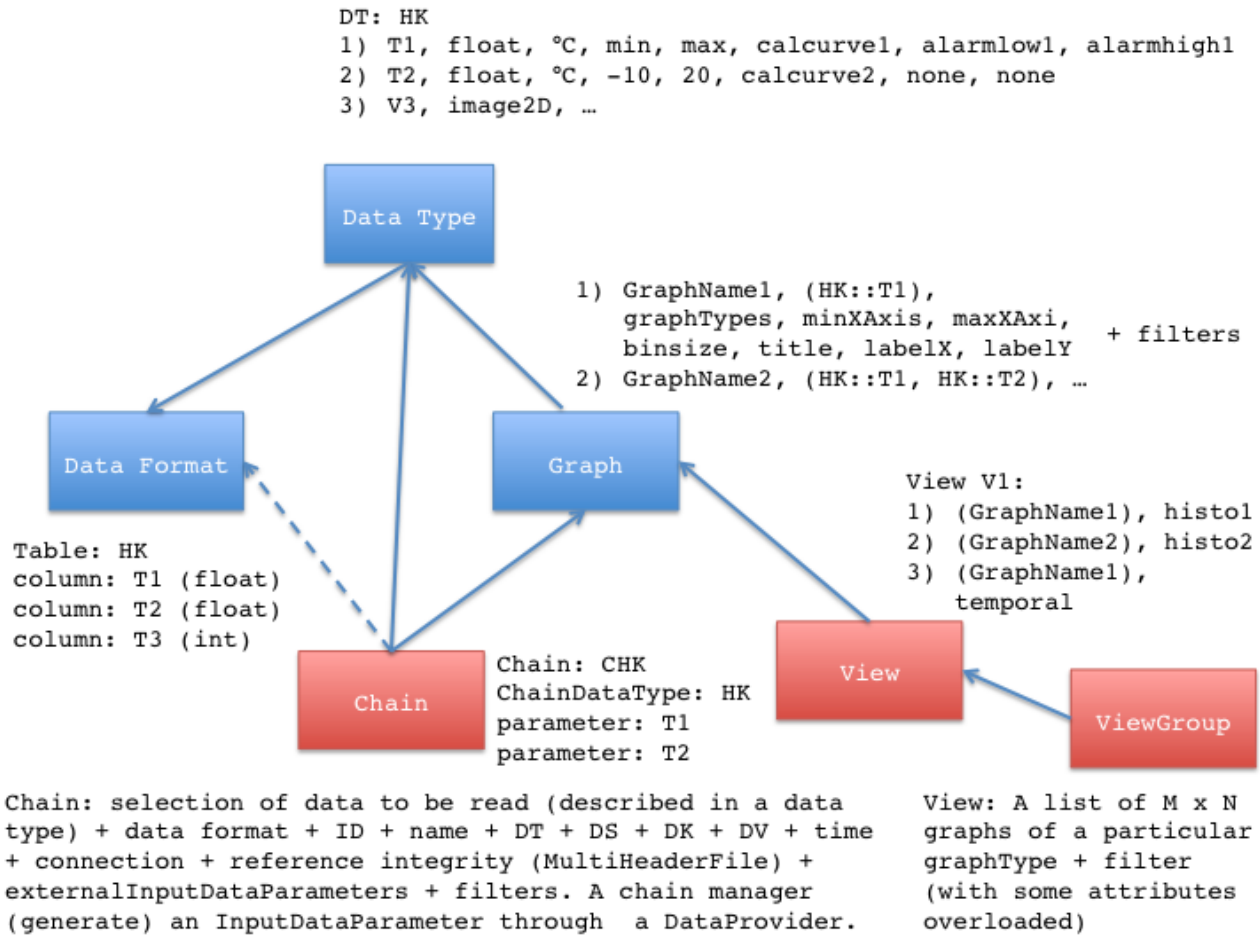

Figure 15: data type, data format, data view, chains.

CIWS-SR-FR-5200 "*QL Views*"

Warning: this requirement contains implementation terminology

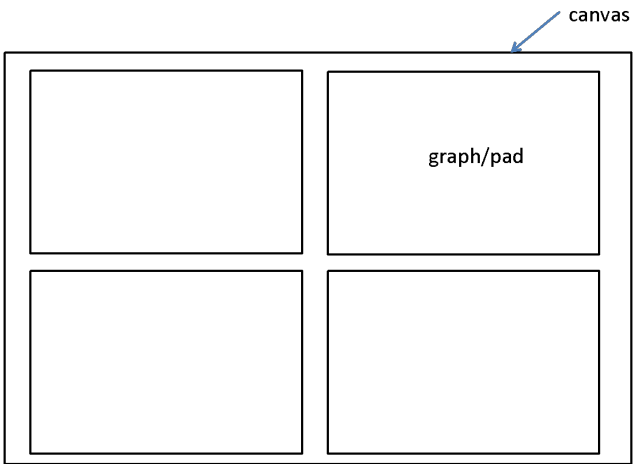

Figure 16: Canvas and pads.

The QL system shall be able to display the fields of a data type (a graph, see **CIWS-SR-FR-5210**) into ***views***.

The QL should manage a list of views.

Using implementation terminology:

1. *A **window** is a canvas + **widgets** (menu, toolbars, buttons).*
2. *A **canvas** contains a view (one-to-one relationship).*
3. *A view is a list of **pads** (at least one pad).*
4. *A pad contains one or more instances of graph.*
5. *An instance of graph displays one or more fields of a data type*

The content of each view should by displayed to the user or saved into an image periodically or at the end of a measure.

CIWS-SR-FR-5210 "*QL Graph*"

Each graph displays one or more fields of a data type. Each field is displayed according with

A. field mapping
B. graph attributes
C. **decorators** (an operation performed on data display at the end of the visualization, e.g. fitting)

A field should have additional elaboration that should be performed

A. at the end of (decorator) of the data display.
B. by user interaction

CIWS-SR-FR-5220 "*QL filtering"*

It shall be possible to filter the data of a field using a query mechanism in on-line and off-line data processing mode. The final result is a selection of a list of events.

CIWS-SR-FR-5230 "*QL decorator"*

A decorator is an operation performed after the display of a graph.

CIWS-SR-FR-5240 "*QL Graph attributes"*

A graph attributes has the following fields:

- graph type
- graph title
- axis label
- axis min limit value
- axis max limit value
- Y min (display)
- Y max (display)

Some attributes depends by the graph type:

- temporal scale (for time graphs)
- number of bins (for histograms)
- fields with error bars

The graph types that should be displayed are the following:

- histograms (1D, 2D, 3D)
- temporal
- synoptic

CIWS-SR-FR-5300 "*QL Functions"*

The software shall provide to the user the following set of functions:

1) Primary functions

2) Manipulation functions

CIWS-SR-FR-5310 *"QL Primary functions"*

The QL primary function shall be the monitoring tasks, primitive accumulations of image/spectral/timing data. These functions comprise:

1) histogram *accumulation* and *display*

3) image display function (2D and 3D)

3) time profile *display*

4) "data field i" vs "data field j" (2D) profile *display*

5) "data field i" vs "data field j" vs "data field k" (3D) (NB: i, j or k should be the time) profile *display*

6) synoptic views with range limits and alarms

7) TBD

Some examples:

The QL shall be able to produce a monodimensional histogram of counts versus either the channel number or the energy values associated to the channel number.

The QL shall be able to display the accumulation of the scientific data as required in order to analyze the functional, performance and calibration tests.

The QL shall be able to produce a monodimensional histogram of counts and/or energy versus time. Each histogram element shall measure the number of detected events or total energy corresponding to the time interval

CIWS-SR-FR-5330 *"QL Input of the primary functions*"

The inputs of the QL primary functions should be the **data types database**. The range, calibration curve and predefined value of monitoring parameters should be stored in the data types archive.

CIWS-SR-FR-5340 *"QL Data comparison"*

Optionally, a comparison of the same data coming from different data products should be allowed.

CIWS-SR-FR-5350 "*QL Manipulation functions"*

The Manipulation function class comprises functions performing data rearrangements, mathematical/algebric operations and more complex numerical analysis using as main input the data on output from the Primary Function or Manipulation Functions themselves. A possible set of functions can be:

1) Integer Rebinning

2) Zoom operations

3) Function fitting

4) Basic statistics

These functions should be performed manually or automatically (attached to a *view*).

CIWS-SR-FR-5360 *"List of fields to display and monitor"*

The QL shall be able to display a list of data type fields (parameters or not), grouped by instrument or telescope subsystem.

CIWS-SR-FR-5370 *"QL monitoring parameters"*

The QL shall inspect and analyze the monitoring parameters in order to perform on them:

A. limit check: checks whether the value is within a specified range
B. delta check: checks whether the rate of change of the value is inside the specified range
C. status check: checks the value of a HK status value against a predefined status value

CIWS-SR-FR-5380 *"QL fields selection display"*

The fields of a data type should be displayed in an alphabetical form and in a graphical form as *time profile accumulation*.

CIWS-SR-FR-5390 *"QL parameter monitoring selection"*

The user should be able to specify the parameters to be monitored.

CIWS-SR-FR-5400 *"QL parameters out of limit check"*

The user shall be notified of the occurrence of any Out Of Limit (OOL) with modalities depending on the parameters to be monitored.

2.1.3 DAS

CIWS-SR-FR-6000 *"Main functionalities"*

a. archiving of the DL0 and DL1 data
b. queries (on metadata) and data retrieval (of data) as result of queries.
c. Data storage of the logging system

CIWS-SR-FR-6020 "*Working mode (basic unit of work)"*

The DAS sub-system shall archive data and related metadata **measure-by-measure**.

CIWS-SR-FR-6030 "*DAS Data Processing mode"*

The DAS system shall work in **off-line mode**.

2.1.4 Reference Catalogues

CIWS-SR-FR-6040 *"Reference Catalogues main functionalities"*
The Reference Catalogues functional unit should provide a set of celestial objects catalogues to QL for a manual check of scientific data.

CIWS-SR-FR-6050 "*Reference Catalogues Data Processing mode"*

The DAS system shall work in **off-line mode**.

2.1.5 Logging system

CIWS-SR-FR-6100 *"Logging system main functionalities"*
The Logging system shall record events occurred during or after the data acquisition to understand the activities performed and the results obtained with the measure. The logs should be edited manually by the user or should be automated.

CIWS-SR-FR-6110 *"Logging system Plan, Session and Measure Block logs"*
The plan, session and measure block log is a manual log where the user records the main purpose of the data taking.

CIWS-SR-FR-6120 *"Logging system Measure log"*

The measure log is a manual log where the user records the main purpose of the measure and the results.

CIWS-SR-FR-6130 *"Logging system Pipeline log"*
The pipeline log is an automated pipeline where each functional unit record events occurred during or after the data acquisition.

CIWS-SR-FR-6140 "*Logging system Data Processing mode"*

The logging system system shall work in **on-line** and **off-line** processing mode.

2.1.6 General requirements of the CIWS-FW

CIWS-SR-FR-7000 *"Common Configuration System"*

The FW and tool must provide a configuration system for manipulating shared configuration, parameters and options for each tool.

CIWS-SR-FR-7010 *"Data Definition Language"*

A DDL shall be provided by FW.

The DDL described

- the data type
- the data format
- the data view (graph, views, group views)

A mapping mechanism between high level data type and data format should be provided by the system.

2.2 Performance requirements

Performance requirements specify numerical values for measurable variables used to define a function (e.g. rate, frequency, capacity, speed and accuracy). Performance requirements may be included in the quantitative statement of each function or included as separate requirements.

CIWS-SR-PER-0010 *"DPS data acquisition rate"*

The DPS data acquisition rate shall be the same of the instrument data generation rate.

CIWS-SR-PER-0020 *"DPS quick-look display rate"*

The DPS/QL display rate shall be the same of the instrument data generation rate. The QL should provide mechanism of data selection (e.g. data analysis mode in sample mode) to be responsive to user interaction.

CIWS-SR-PER-0030 *"DPS quick-look scalability and multiplatform"*

The QL should be a scalable, distributed and multiplatform solution.

2.3 Interface requirements

Interface requirements specify hardware, software or database elements that the system, or system component, must interact or communicate with. Accordingly, interface requirements should be classified into software, hardware and communications interfaces. Interface requirements should also be classified into internal and external interface requirements, depending upon whether or not the interface coincides with the system boundary. An interface requirement may be stated by:

- *describing the data flow or control flow across the interface*
- *describing the protocol that governs the exchanges across the interface*
- *referencing the specification of the component in another system that is to be used and describing:*
  - *when the external function is utilized*
  - *what is transferred when the external function is utilized*

2.4 Operational requirements

Operational requirements specify how the system will run (i.e. when it is to be operated) and how it will communicate with human operators.

CIWS-SR-OR-0010 *"Human operations"*

The functionalities of the IW system (data acquisition and archiving, quick look) shall be operated by a single operator.

CIWS-SR-OR-0020 *"Scriptability"*

The FW shall be able to be operated using a scripting language to specify the order in which pipeline tools are run, configuration options, and data flow.

CIWS-SR-OR-0030 *"Independent tools"*

It shall be possible to run individual tools independently of the pipeline software

2.5 Resource requirements

Resource requirements specify the upper limits on physical resources such as processing power, main memory, disk space,…

2.6 Verification requirements

Verification requirements constrain the design of the product. They may do this by requiring features that facilitate verification of system functions, or by saying how the product is to be verified. Verification requirements may include specification of any:

- *simulations to be performed*
- *requirements imposed by the test environment*
- *diagnostic facilities*

CIWS-SR-VR-0010 *"Unit test library"*

The FW will provide or support a common unit testing library or convention that is used by all tools.

CIWS-SR-VR-0020 *"Unit Testing "*

All pipeline tools, frameworks, and libraries shall have unit tests for all major functionality that can be run automatically to ensure the correct working of each component

2.7 Acceptance testing requirements

CIWS-SR-AR-0010 *"Regression Test Library"*

The FW shall provide a common library or scripting facility for performing regression tests.

CIWS-SR-AR-0020 *"Regression tests"*

The FW shall have a set of well-defined regression tests that can be used to verify the stability of results when changes are made to individual parts of the software. This will help for evaluating the stability of the system and for cross-checking different algorithms and for interfacesverification.

CIWS-SR-AR-0030 *"Automated Regression Testing"*

A facility for the automation of regression testing will be provided by the FW. Useful for checking the stability of software with any change in code or external information

2.8 Documentation requirements

CIWS-SR-DR-0010 *"Code Documentation System"*

A common code documentation system shall be used in all tools and libraries. The FW should provide a template to comment each element of the code (class, methods, attributes, etc.).

CIWS-SR-DR-0020 *"Tool Documentation"*

Each tool will also include documentation describing the general operation of the tool, including a description of the algorithm(s) used, the intended inputs and outputs, and any external interfaces.

2.9 Security requirements

2.10 Portability requirements

Portability requirements specify how easy it should be to move the software from one environment to another.

CIWS-SR-PR-0010 *"Installable system"*

All components and dependencies of the software must be installable and update-able in a semi-automatic fashion (e.g. using a package system).

2.11 Quality requirements

Quality requirements specify the attributes of the software that make it fit for its purpose. The major quality attributes of reliability, maintainability and safety should always be stated separately. Where appropriate, software quality attributes should be specified in measurable terms (i.e. with the use of metrics).

CIWS-SR-QR-0010 *"Preferred Coding Style"*

FW components, libraries and tool should follow common style guidelines for: commenting, function and variable naming, general design philosophy (to be defined in a separate document)

CIWS-SR-QR-0020 *"Periodic Code Reviews"*

The designs and code should be reviewed periodically by the pipeline or tool maintainer or other experts.

CIWS-SR-QR-0030 *"Version control"*

Software used for data processing and analysis shall be fully version controlled.

CIWS-SR-QR-0040 *"Reproducibility"*

The data processing and results must be reproducible. The software version data shall be stored and managed.

CIWS-SR-QR-0050 *"Modularity"*

Software must be modular, separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality.

CIWS-SR-QR-0060 *"Software framework"*

The software framework provided to IW develop should be minimal in terms of complexity, but fully featured.

2.12 Reliability requirements

Software reliability is 'the ability of a system or component to perform its required functions under stated conditions for a specified period of time' [Ref. 2]. The reliability metric, 'Mean Time Between Failure' (MTBF), measures reliability according to this definition.

CIWS-SR-RR-0010 *"Availability"*

The availability of the IW DPS (acquisition and processing) during data taking must be >99,5 %.

CIWS-SR-RR-0020 *"Availability of data archiving"*

The availability of the IW data archiving (DAS) system must be >98 %.

CIWS-SR-RR-0030 *"Performance monitoring"*

IW sub-systems shall provide performance monitoring information, which can be used to identify emerging problems and ensure the reliable functioning of the system.

CIWS-SR-RR-0040 *"Automated logging"*

Comprehensive logging of conditions, actions, errors and warnings shall take place to ensure that problems and failures can be analysed and avoided in the future.

CIWS-SR-RR-0050 *"Status information"*

Information on the status of sub-systems shall be provided in a readily comprehensible manner to IW operators, to ensure that problems and failures are rapidly identified also during the data acquisition.

CIWS-SR-RR-0060 *"Backups"*

Appropriate planned action should be taken to ensure that the probability of complete loss of a significant fraction of acquired data is low, for example through storage of multiple copies of observation data. The probability of a total loss of acquired data of one day of data taking is < 0.01 %.

2.13 Maintainability requirements

Maintainability is 'the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment'

CIWS-SR-MR-0010 *"Failure of data acquisition and archiving"*

In case of failures, corrective maintenance shall be initiated to restore data acquisition and archiving to its specification or to restore a failed element to operational status. The objective must be to restore the element to satisfactory operation in the shortest possible time, typically within 5-10 minutes.

CIWS-SR-MR-0020 *"Human intervention for corrective maintenance"*

A single person shall operate the corrective maintenance operations of the IW system.

2.14 Safety requirements

Safety requirements specify any requirements to reduce the possibility of damage that can follow from software failure.

CIWS-SR-SAR-0010 *"ICT protection"*

The ICT systems shall have an appropriate level of protection against unauthorised access/hacking.

3.

System design

3.1 Design method

Describe or reference the design method used.

The developer shall construct a 'physical model', which describes the design of the software using implementation terminology using a top-down approach. For each component the following information shall be detailed in the ADD:

- data input;
- functions to be performed;
- data output.

Data structures that interface components and the control flow between the components are defined in this section.

The design method is based on 4+1 Architectural View model already described in Section 2.1. In particular, this Section cover the Implementation View, that illustrates a system from a programmer's perspective and is concerned with software management. UML Component and Package diagrams are used in this context.

This section covers

a) implementation view
b) process view
c) data model domain decomposition

3.2 Data Model Domain decomposition

The E/R diagram shown in **Figure 7** reports the data model from a logical point of view.

From a implementation point of view this data model is divided into

- the relational data model shown in Figure 17 (for the DPS sub-system).
- the data type and data formats, the parameters to be monitored are described by means of XML Schema using XSD.

3.2.1 DPS Relational Schema

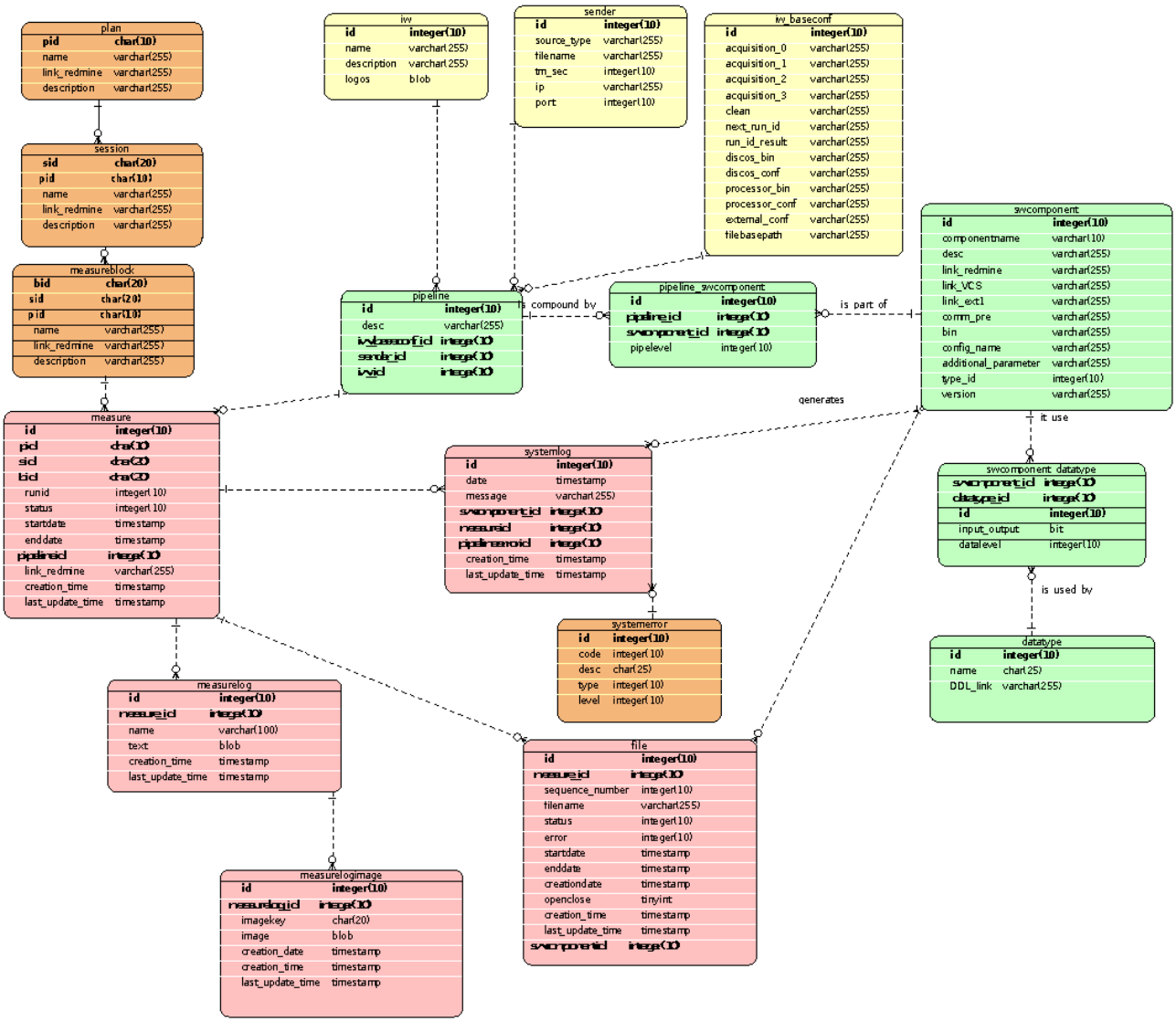

Figure 17: DPS relational schema

3.3 Implementation View

The system is logically divided into layers:

1. The **User Interface** layer, which contains all the boundary classes of the system used by users;
2. The **Application Logic** layer, which contains all the classes that implement the functionality of the system. This layer should be divided into
   2.1. Data **flow** layer
   2.2. Data **transformation** (format/type) layer

3. The **Data Management** layer, which provides all the functionalities used for data storage and retrieval;

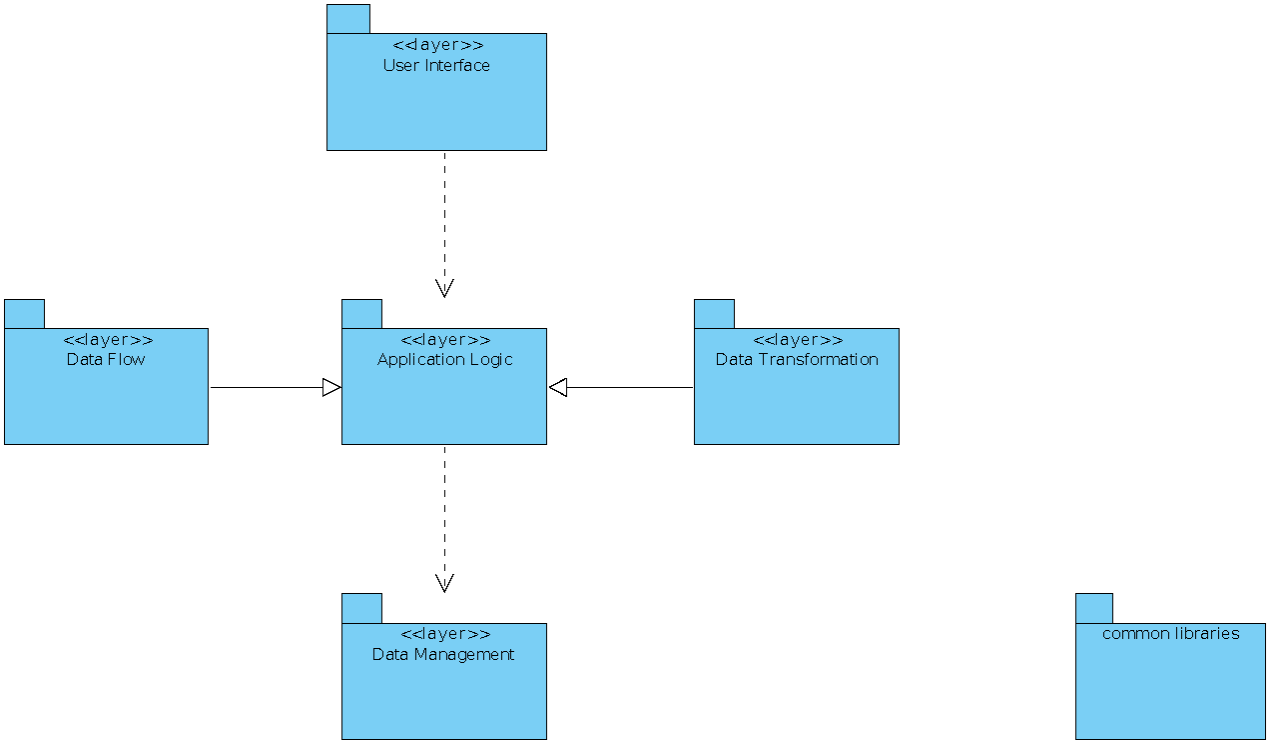

Figure 18: CIWS-FW Layers

Il CIWS-FW è composto da diversi componenti (si veda la Figura) che possono essere così raggruppati:

- DPS: data acquisition (mediante DISCOS o MCS), processing (utilizzando PacketLib e ProcessorLib) e quick-look dei dati. Il sistema fornisce anche componenti di supporto, quali un control software per la configurazione e l'esecuzione della pipeline ed una GUI per visualizzare i log della pipeline
- DAS: data archiving and retrieval
- Reference Catalogues: cataloghi di riferimento, da utilizzare ad esempio nel quick-look
- Data Model: contiene la definizione XML dei data format, data type, data view del CIWS
- Common Libraries:
    - DPS: librerie comuni fornite dal DPS
    - DAS: librerie comuni fornite dal DAS
- External libraries: dipendenze esterne

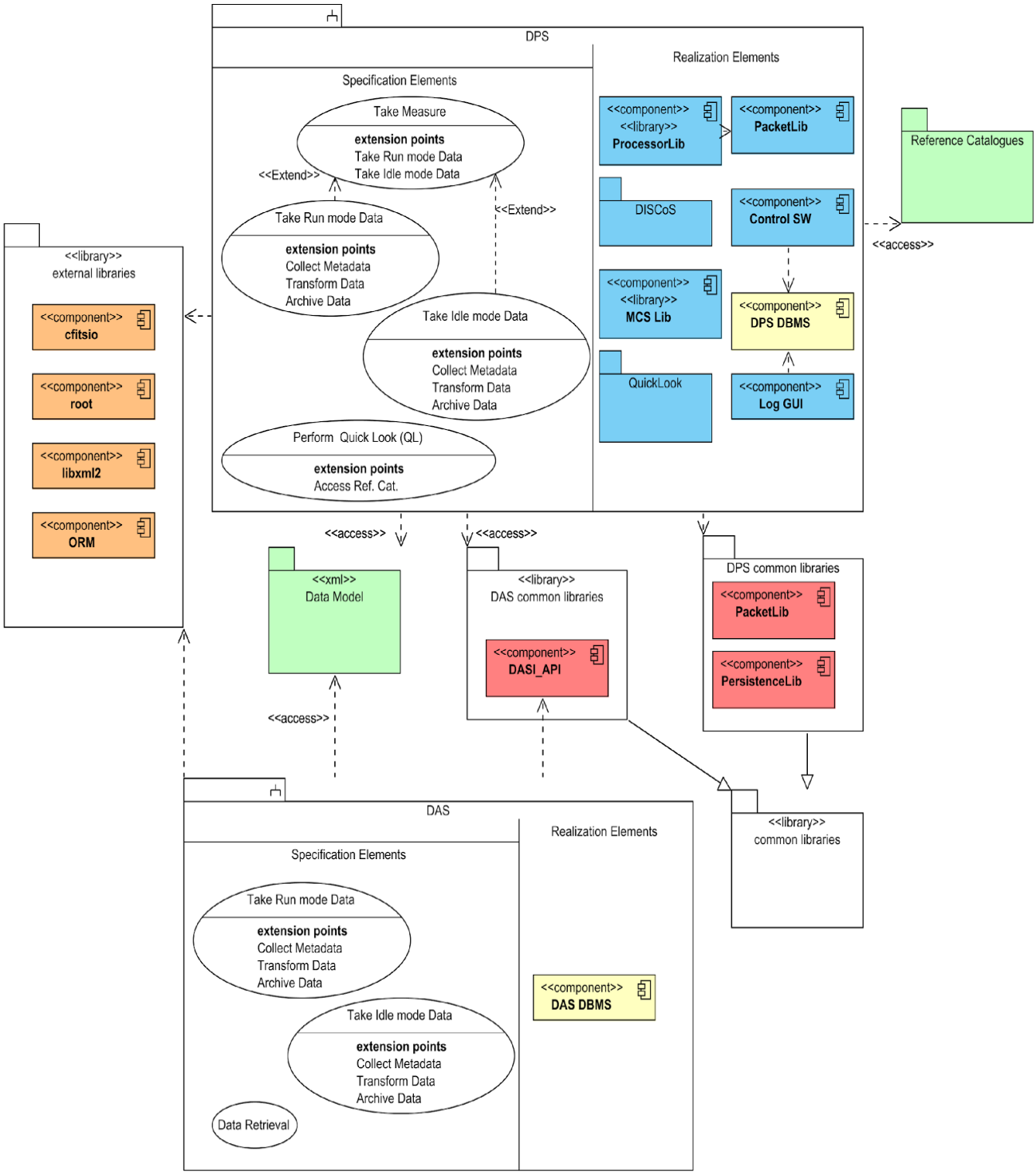

Figure 19 Architettura generale del CIWS-FW

Nella seguente figura è riportato lo schema generale dell'architettura del DPS, con le dipendenze e le relazioni esterne.

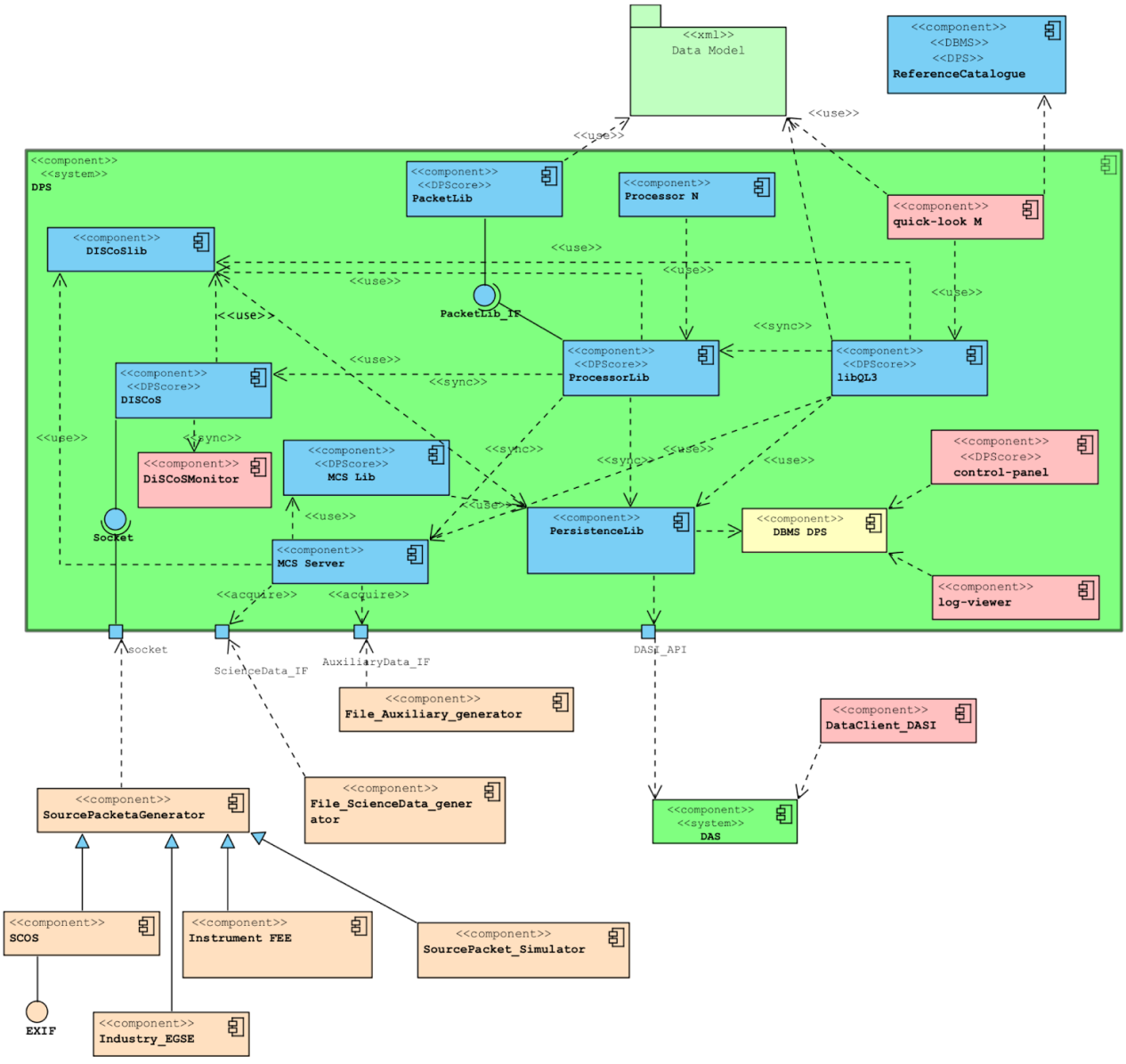

Figure 20

Il DPS, il DAS e i reference catalogues possono essere visti come sistemi indipendenti ma che possono essere collegati mediante opportune interfacce.

I componenti di data acquisition del DPS sono

- **DISCOS**: per l'acquisizione di un flusso di source packet, l'archiviazione su file system e i meccanismo di sincronismo con i processori (identificati con lo stereotipo <<sync>>) mediante la **DISCOSlib**)
- **MCS Server**, componente di acquisizione di files, sviluppato utilizzando la **MCS Lib**

I **sistemi esterni** al CIWS sono evidenziati come componenti arancio nella figura, in particolare
- source packet generator: genera i source packet per il sistema DISCOS: le sorgenti dati possono essere diverse, ad esempio l'EGSE fornito dall'industria, SCOS, la FEE dello strumento, un simulatore software di source packets
- File generator: il flusso dati può essere rappresentato da file (di dati scientifici o ausiliari) che sono acquisti dall'MCS server

Alcune librerie di base consentono il processamento e la conversione dei source packets in un diverso formato file:

- **PacketLib**: libreria C++ che permette la codifica, decodifica e routing dei source packet. Questa libreria è utilizzata dalla ProcessorLib

- **ProcessorLib**: libreria utilizzata per la scrittura dei **Processori** (convertitori da source packet a file di output in diverso formato). Fornisce l'interfaccia verso DISCOS e permette allo sviluppatore che utilizza il CIWS-FW di concentrarsi solo sulla scrittura dei **Processori** stessi, lasciando alla ProcessorLib il reperimento da DISCOS e il routing dei source packet, il meccanismo di sincronismo con DISCOS, l'interfaccia con il quick-look, e l'interfaccia con la **PersistenceLib**. Il formato di output è invece lasciato a scelta dell'utente. Formati utilizzati dal CIWS-FW sono FITS e XML.

- **PersistenceLib**: libreria utilizzata dai diversi componenti del CIWS-FW per la memorizzazione nel database DPS dei path dei file di dati acquisiti e dello stato della pipeline. I dati sono memorizzati nel **DBMS DPS**

- **libQL3**: libreria utilizzata per lo sviluppo dei **quick-look**

Altri componenti del sistema sono invece forniti di interfaccia grafica utente:

- **DISCOS Monitor**: un monitor che permette di controllare l'acquisizione dei source packets e il passaggio di questi ai vari processori

- **Control panel**. Un componente che permette l'esecuzione della pipeline del DPS e la modifica dei parametri della pipeline stessa (mediante interfaccia web)

- **Log viewer:** interfaccia web che permette l'interrogazione dei dati memorizzati dalla PersistenceLib nel DBMS DPS

- **Quick-look**: componente per la visualizzazione grafica dei dati acquisiti, già descritti in questa sezione

3.3.1 DPS processing measure

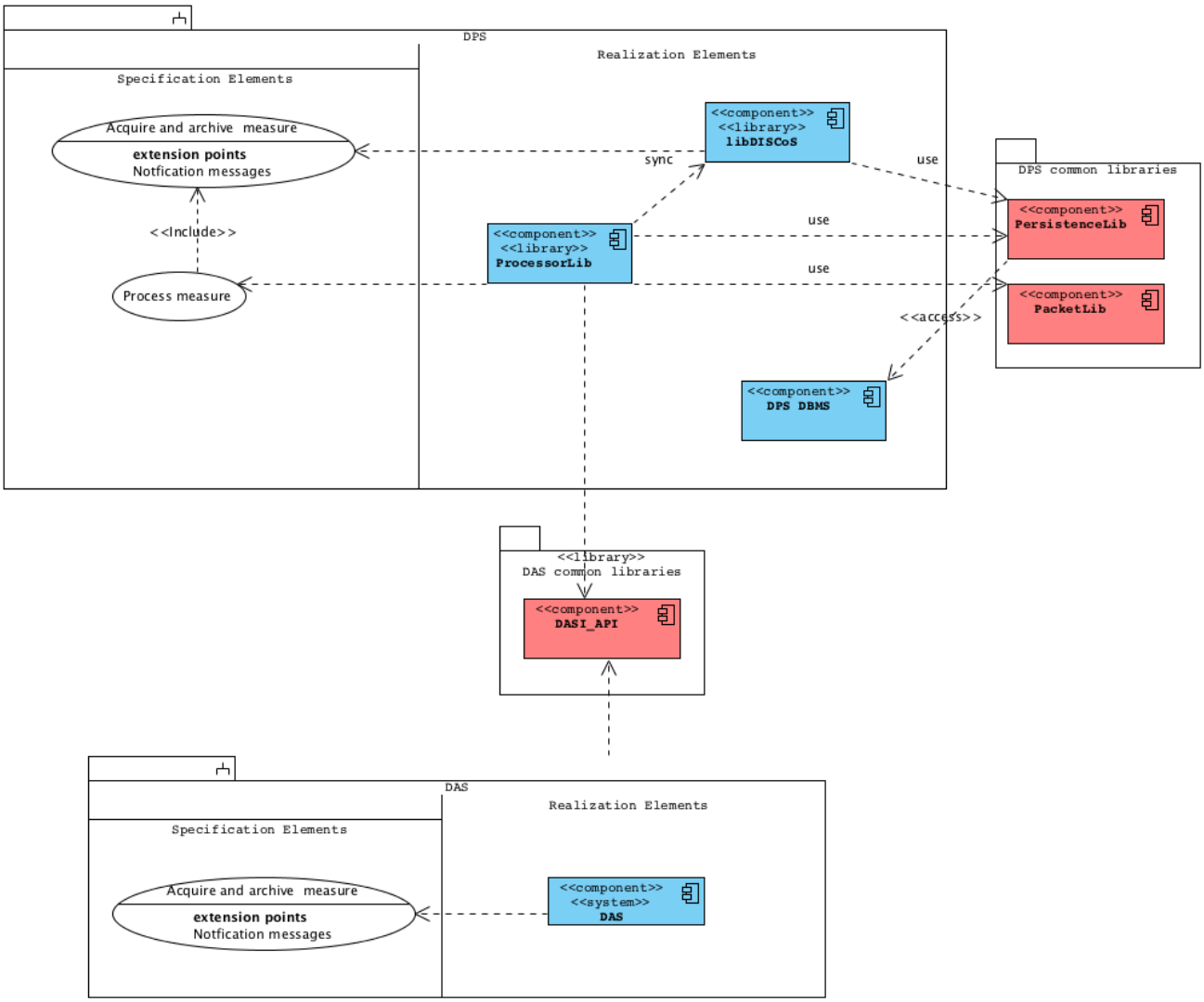

Figure 21

3.3.2 QL Package diagram

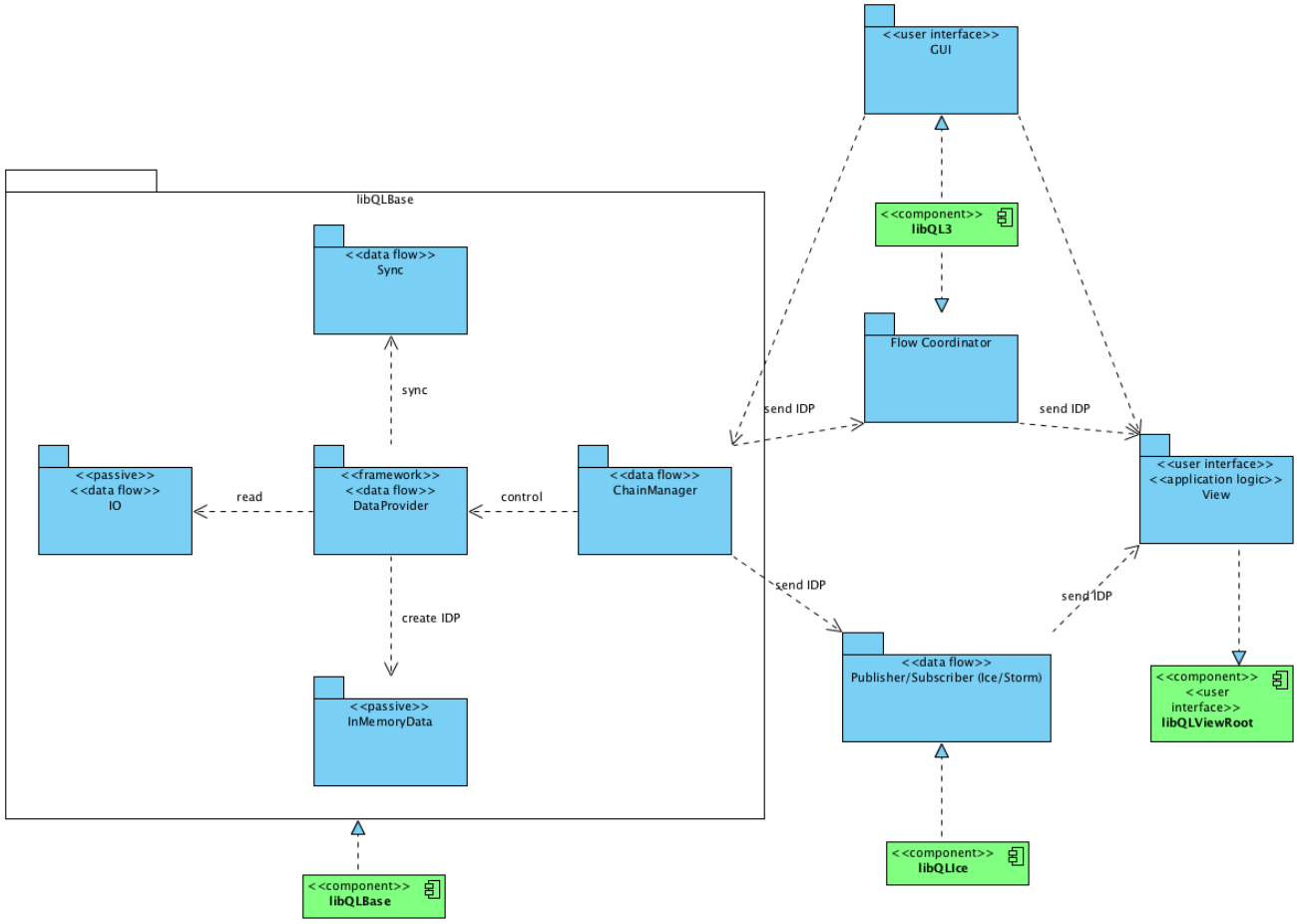

Figure 22

3.4 Process View

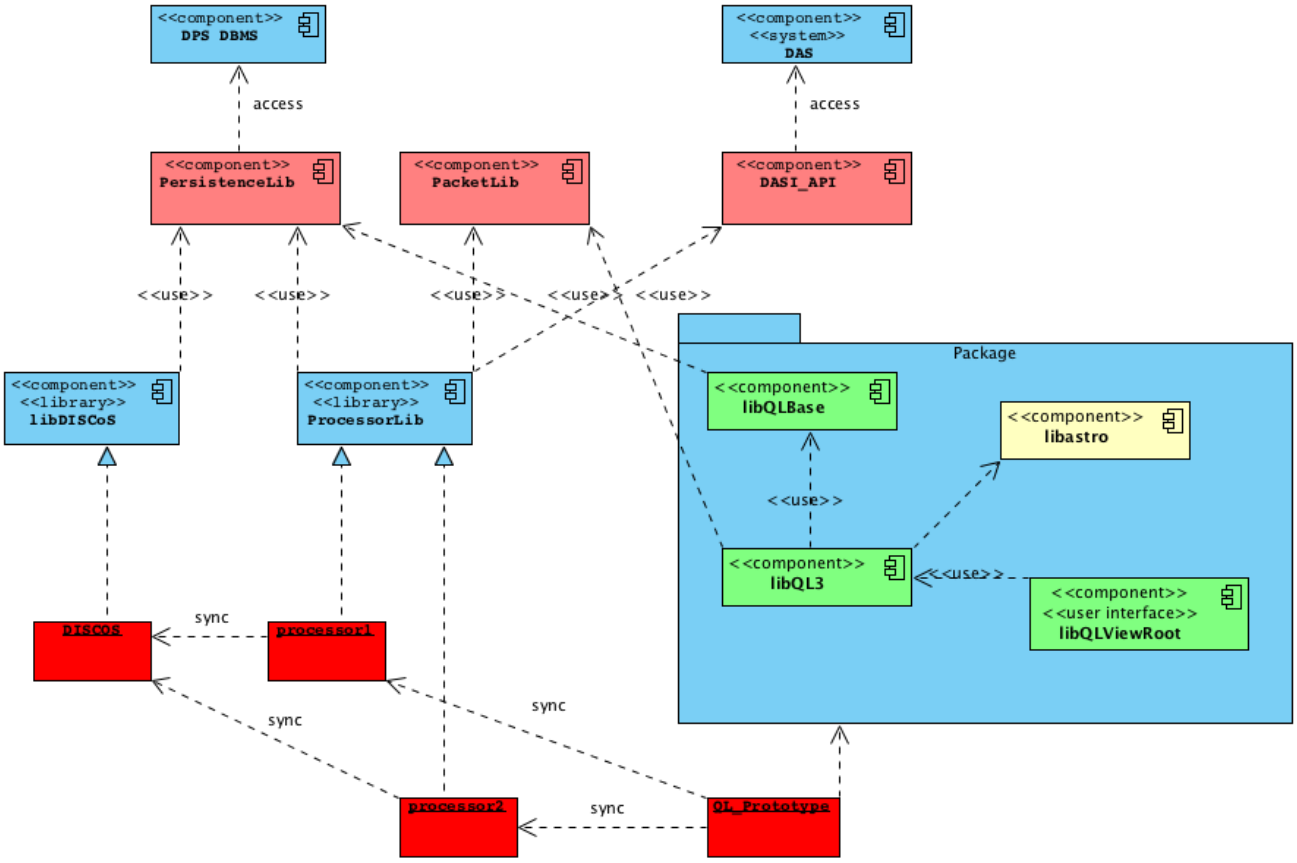

Figure 23: process view of the CIWS system