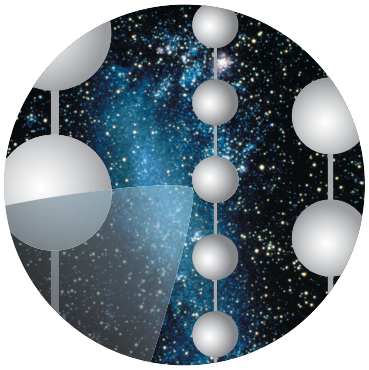


IceCube Software

The Software Design Description of "Production" IceCube Software

Simon Patton L.B.N.L.

Last Modified: February 18, 2003 2:23 pm

Version:0.3

1.0 Introduction

1.1 Purpose

The purpose of this document is provide full dscription of the IceCube "production" software system architecture and document how this is realized by the system design. In this context "production" software is defined as those systems and components which contribute to the successful realization of the vision for IceCube software.

> To provide the physicist, in collaboration with the DAQ hardware, with stable, reliable, reproducible access to data taken by the IceCube detector and data created by any approved simulation of the detector, so that they can publish physics papers.

The intended audience of this document is anyone who plans to contribute to the "production" software. This should cover both software developers and software managers on the IceCube project. This document may also help physicists, who aim to derive physics result from IceCube data, to understand how their data is handled before they gain access to it.

1.2 Scope

As outlined in the IceCube software vision above, the scope of "production" software, and thus this document, is all software that is used to:

- operate the IceCube detector;
- read data from the detector and record and archive data which is deemed to be important for generating physics results;
- transfer of this recorded data from the South Pole to the "Northern Hemisphere";
- store, catalog and process with approved algorithms, this data in a Data Warehouse;
- provide, upon request from a physicists, data for analysis in suitable data formats.

It also covers a similar chain of software for "any approved simulation of the detector".

What "production" software does not explicitly cover is any physicist's individual analysis software including, but not limited to:

- analysis specific processing of the data;
- generation of private simulations of the data;
- preparation of physics papers for publication[1].

That does not mean that individual analysis can not make use of any of the components developed as part of the "production" software, it simply means that such analyses are not necessarily considered to be the primary clients of such components.

1. While the software used in the preparation of physics papers for publication may be common to all members of the collaboration, the requirements for such software are distinct and separate enough for it not to be dealt with in this document, but rather have a document of its own.

1.3 Definitions, acronyms and abbreviations

design entity: (IEEE Std 1016-1998) An element (component) of a design that is structurally and functionally distinct from other elements and that s separately named and referenced.

design view: (IEEE Std 1016-1998) A subset of design entity attribute information that is specifically suited to the needs of a software project activity.

1.4 References

The following references are used throughout this document.

[IEEE Std 610.12-1990] Standard Glossary of Software Engineering Terminology.

[IEEE Std 1016-1998] Recommended Practice for Software Design Descriptions.

1.5 Overview of this document

The next section of this document review the architectures that have been chosen in to construction of the IceCube software system. The rest of this document is laid out along the line specified in IEEE Std 1016-1998, namely it describes the design using four different design views; Decomposition, Dependencies; Interfaces and Details. In each view all of the design entities that are components of the “production” software are detailed with those attributes relevant to that view.

2.0 Architectural Overview

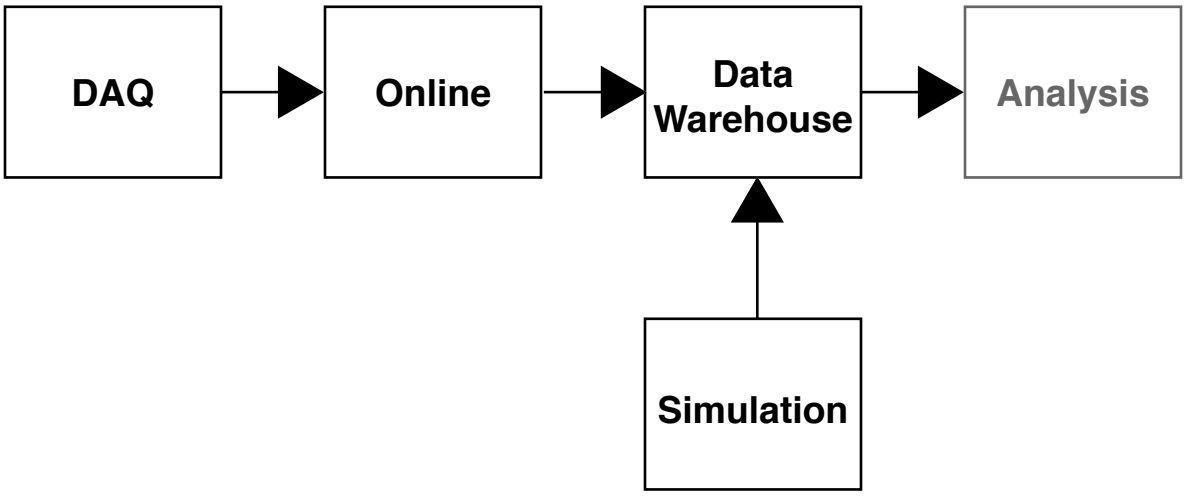
To facilitate the understanding of the IceCube software system design it helps to take a look at the architecture on which is it based. This section provides an overview of this architecture and explains some of the reason behind its development.

2.1 Dominant Architecture

If we review the vision of the IceCube software we can see that its core responsibility is to move data from the detector and deliver it to physicists for analysis. Figure 1 illustrates this data flow. Data from the DOMs in and on the ice are brought together by the DAQ where the trigger decides which of these data may be required for physics analysis and these data are forwarded to Online where a closer inspection of the data is made. Data which is deemed to be "urgent" for analysis is transferred to the "northern hemisphere" over a satellite link while the rest of the data is stored at the South Pole to be transferred at a later data. (All data output by the DAQ is archived at the South Pole.) Once the data has arrived in the northern hemisphere, whether by satellite or other means, is stored in a Data Warehouse and is made available for physicists to analyze. To help provide consistent data for analyses, data can be processed by a standard set of algorithms as it enters the Data Warehouse.

Figure 1 also shows that simulated data can be created in parallel to that taken at the South Pole. This simulated data is also delivered to the Data Warehouse for standard processing and storage.

FIGURE 1: The main flow of data through the IceCube system.

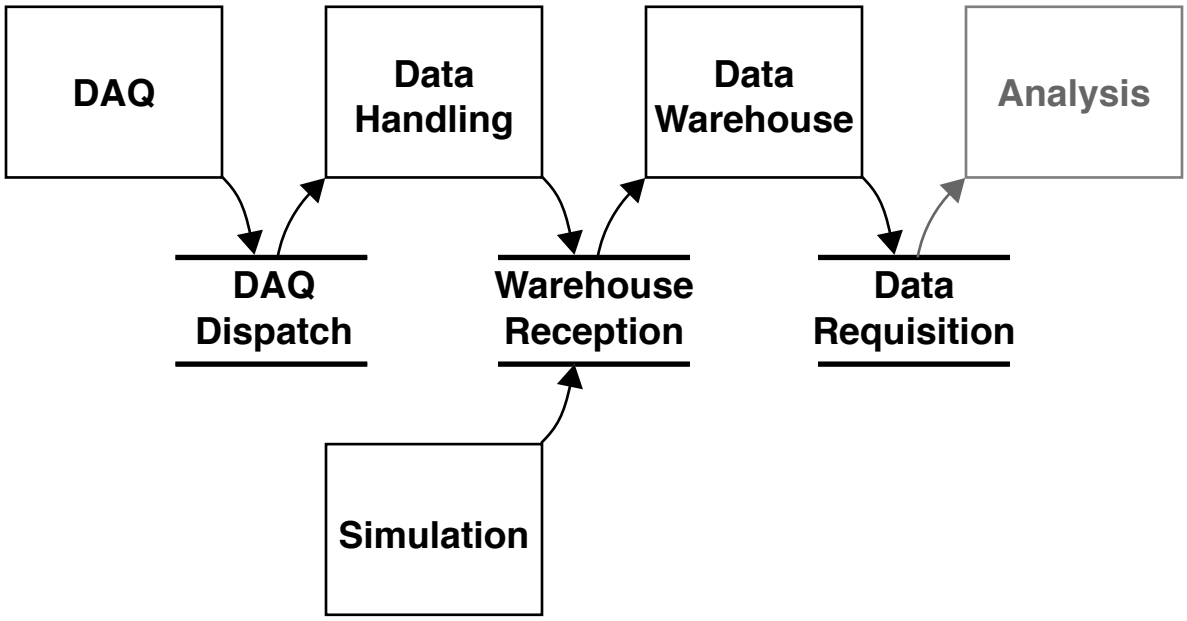


Given the linear nature of the dataflow a pipeline, or pipe and filter as it is also known, architecture is a natural choice for the system designed to handle this flow. This is not only true at the system level but also most of the subsystems architectures will follow this pattern.In the pipeline architectural pattern a stream of data is processed, sequentially by a series of entities, often called filters, and the data itself is "piped" between each of these entities in turn. Thus a sequence of processing entities is called a pipeline.We can now look at the overall system in terms of pipes and filters. Figure 2 shows how the main top level design entities collaborate to form the basic pipeline. In this figure Online has taken on its new name of Data Handling as "Online" more commonly used to refer to just the processing of data immediately after acquisition and does not include transport of data way from the detector as was the original case here.

Figure 2 also illustrates that the major subsystems; DAQ, Data Handling, Simulation and the Data Warehouse, are functionally independent of each other, i.e. the can run without any of the other major subsystems being present, while the minor subsystems; DAQ Dispatch, Warehouse Reception and Data Requisition, act as the pipes between the major subsystem, providing data buffering and removing any dependency between the major subsystems. Figure 3, which shows the dependencies between the high level design elements, illustrates how direct dependencies between major subsystems is avoid by using shared minor subsystems. This independence means that during development and test of the major subsystems stub implementation of the minor subsystems allow for parallel development.

Figure 3 also introduces two more of the top level design entities, South Pole apparatus Management and IceBucket and absorbs the DAQ into the more general Detector Operations subsystem. This first of the new subsystems is the subsystem that is resposibible for managing all of the software and hardware infrastructure physically located at the South Pole. This combines the management of the DAQ with the management of part of the Data Handling into a single unified interface for the convenience of the operator sta-

FIGURE 2: The data flow through the main top design entities of the IceCube software systtem.



tioned at the South Pole. The second new subsytem, IceBucket, is a collection of utility codes that provide implementations of common functionality. This common functionality is discussed further in the next section.

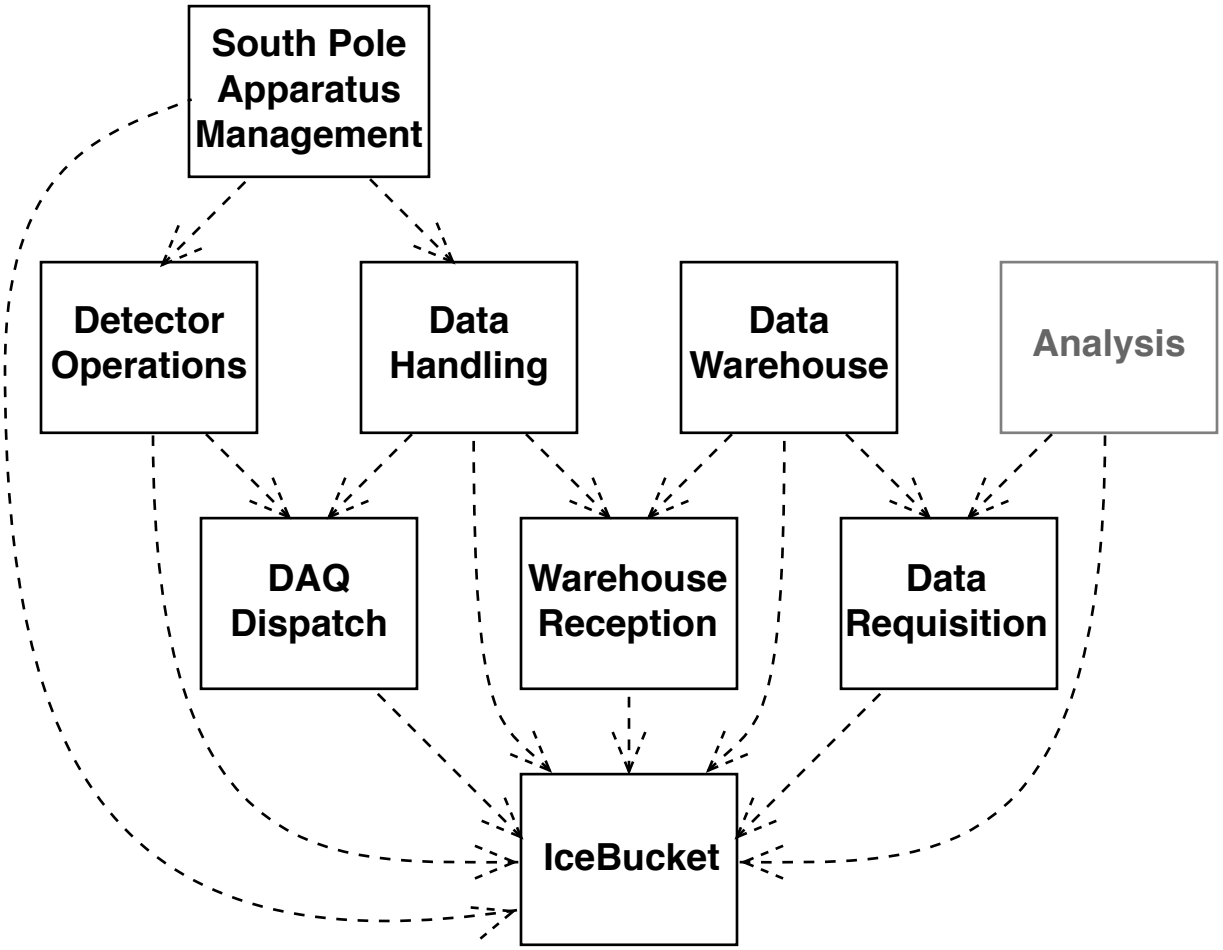
2.2 Subordinate Architectures

While the dominant architecture is derived fromm the main dataflow of the experiment, there are a number of subordinate architectures that recure in some or all of the subsystems. While the pipeline architecture allows for the major subsystem to be developed in parallel that does not mean that they should be developing in ignorance of the others. Indeed, all the major subsystems have similar needs across various aspects of the operation and using common architectures as solutions to these problems is very beneficial. Not only is there a reduction in the resources needed to develop a solution if it is only developed once, but it also allows, where necessary, for intergration of these aspects across subsystems. An example of this is the "South Pole Apparatus Management" subsystem which needs to combine aspects of both DAQ and Datahandling. Another example is Data Transfer which must take place both inside and between subsystems so a single solution makes sense.

The major subsystems have the following aspects in common with each other:

- Data Transfer;

FIGURE 3: The dependencies of the top level design entities.



- Control, including user interfaces and application configuration;
- Monitoring; and
- Logging.

All these aspects are contained, in some form or another, the in IceBucket utility collection. There basic architecture is briefly discussed here to give subsystem developers of the support available by utilizing IceBucket.

2.2.1 Data Transfer

2.2.2 Control

2.2.3 Monitoring

2.2.4 Logging

3.0 Decomposition

This section describes the division of the software system into design entries.

ID: Production Software

Type: System

Purpose: To provide the physicist, in collaboration with the DAQ system, with stable, reliable, reproducible access to data taken by the IceCube detector and data created by any approved simulation of the detector, so that they can publish physics papers.

Function: The production software covers the management of both the detector and data, whether real or simulated, so that physics data can be provided to the physicist upon request. To achieve this end the system is responsible for the following.

- Control and monitoring of the detector at the South Pole.
- Acquisition of all data from the detector hardware necessary to generate useful physics data, this can include, but is not limited to, raw data from the DOMs, filtered data from the DOMs, any calibration data, and local environmental data.
- Archiving of all recorded data at the South Pole and managing the data's transfer to the Northern Hemisphere Data Warehouse.
- Management of the Northern Hemisphere Data Warehouse.
- Creation of simulated data sets for engineering and physics studies and their transfer to the Northern Hemisphere Data Warehouse.
- Providing the physicists with a set of data files from the Data Warehouse, that are a response to a specific query.

Subordinates:

- South Pole Apparatus Management
- Detector Operations
- DAQ Dispatch
- Data Handling
- Simulation
- Warehouse Reception

- Data Warehouse

3.1 South Pole Apparatus Management

ID: South Pole Apparatus Management

Type: Minor Subsystem

Purpose: To manage all aspects of the hardware and software phyicially located as the South Pole by coordinating the Detector Operations, DAQ Dispatch and Data Handling subsytems.

Function: This subsystem is responsible for presenting a unified interface, to an operator, so that they can manage all of the systems located at the South Pole. To do this it must perform tasks which include, but are not limited to, the following:

- Present the Control and management interfaces of the Detector Operations, DAQ Dispatch and South Pole portion of the Data Handling subsystems to the operator in a unified fashion .
- Coordinate all tasks that span the Detector Operations, DAQ Dispatch and Data Handling subsytems.

Subordinates:

3.2 Detector Operations

ID: Detector Operations

Type: Major Subsystem

Purpose: To manage all aspects of the detector at the South Pole and provide to DAQ Dispatch any data generated by the detector that is needed to do physics analysis.

Function: This subsystem is responsible all aspects of detector operations. To do this it must perform tasks which include, but are not limited to, the following.

- Control and management of the detector hardware.
- Configuration management of the DAQ Software system.
- Monitor both the hardware and software portions of detector.
- Acquire all the data from the detector hardware and decide which data should be kept and which shouldn't.
- Format all kept data and transfer it to DAQ Dispatch.
- Execute all tests needed during the commissioning and running of the detector.

Subordinates:

- Experiment Control
- Detector Configuration
- Data Acquisition
- Real Time Monitoring

- Fabrication and Commissioning

3.3 DAQ Dispatch

ID: DAQ Dispatch

Type: Minor Subsystem

Purpose: To act as a repository for data produced by the DAQ.

Function:

- Receive new data from DAQ.
- Signal Data Handing that new data exists.
- Transfer data to the Data Handling when requested.

Subordinates: TBD

3.4 Data Handling

ID: Data Handling

Type: Major Subsystem

Purpose: To manage the movement of all data output by the DAQ system, and any other systems at the South Pole, to the Data Warehouse in the Northern Hemisphere.

Function: This subsystem is responsible for the handling of all data taken at the South Pole. To do this it must perform tasks which include, but are not limited to, the following:

- Monitor DAQ Dispatch and identify new data.
- Request transfer of new data from DAQ Dispatch into this subsystem.
- Archive of all data on to tertiary media.
- Transfer the archived data to Warehouse Reception once the tertiary media has been transported up to the Norther Hemisphere.
- Categorize Events produced by the DAQ system to those that should be transferred over the satellite link, and those which shouldn't.
- Transfer data, deemed to be important, over the satellite link and, when it arrives in the Northern hemisphere, transferring this data to the Warehouse Reception.

Subordinates:

- South Pole Archiving.
- Online filters.
- Satellite Transfer.

3.5 Simulation

ID: Simulation

Type: Major Subsystem

Purpose: To generate simulated data for all aspects IceCube data so that it can be used for engineering and physics studies.

Function: To support both engineering and physics studies this subsystem must perform tasks which include, but are not limited to, the following:

- Generation of physics events.
- Model the propagation of any particles generated in those events from their point of origin to their creation of photons.
- Model the propagation of the photons from their point of creation to their point of detection, if they have one.
- Simulate the response of the DOM to one or more incident photons, and well as the DOM response for no incident photons, i.e. noise.
- Propagate any data created by the DOM through a simulation of the DAQ and online filtering subsystems to create a complete simulated data set.
- Transfer the resultant data set to the Warehouse Reception.

Subordinates:

- Generation.
- Particle Propagation
- Photon Propagation
- Detector Simulation
- Online Simulation

3.6 Warehouse Reception

ID: Warehouse Reception

Type: Minor Subsystem

Purpose: To act as a repository for all data that is being added to the Data Warehouse.

Function:

- Receive new data from either Data Handling or Simulation.
- Signal the Data Warehouse that new data exists.
- Transfer data to the Data Warehouse when requested.

Subordinates: TBD

3.7 Data Warehouse

ID: Data Warehouse

Type: Major Subsystem

Purpose: To store the data recorded by the IceCube detector and data generated by approved simulations.

Function:

- Monitor the Warehouse Reception and identify new data.
- Request transfer of new data from the Warehouse Reception into this subsystem and entering any necessary metadata into the Warehouse's catalog.
- Process new data, and when requested re-process existing data, with approved reconstruction algorithms, storing the resulting data into this subsystem and entering any necessary metadata into the Warehouse's catalog.
- Monitor the Data Requisition and identify new queries.
- Request transfer of any new query from the Data Requisition.
- Create a set of files that satisfy the query.
- Transfer the set of files to Data Requisition which are the response to the query.

Subordinates: TBD

3.8 Data Requisition

ID: Data Requisition

Type: Minor Subsystem

Purpose: To provide an interface through which physicists can request data based on a provided query.

Function:

- Receive new queries for physicists.
- Signal the Data Warehouse that a new query exists.
- Transfer query to the Data Warehouse when requested.
- Receive a set of files from the Data Warehouse which are the response to the query.
- Signal to the physicist that the response to their query is ready.

Subordinates: TBD

4.0 Dependencies

4.1 South Pole Apparatus Management

ID: South Pole Apparatus Management

Type: Minor Subsystem

Dependencies:

- Detector Operations - .

- DAQ Dispatch - .
- Data Handling - .

Resources:

-

4.2 Detector Operations

ID: Detector Operations

Type: Major Subsystem

Dependencies:

- DAQ Dispatch - the destination for all data recorded form the detector.

Resources:

- DAQ CPU Farm.

4.3 DAQ Dispatch

ID: DAQ Dispatch

Type: Minor Subsystem

Dependencies: None

Resources:

- DAQ Dispatch disk farm.

4.4 Data Handling

ID: Data Handling

Type: Major Subsystem

Dependencies:

- DAQ Dispatch - the source of all data recorded at the South Pole.
- Warehouse Reception - the destination of all data recorded at the South Pole.

Resources:

- Online CPU Farm.

4.5 Simulation

ID: Simulation

Type: Major Subsystem

Dependencies:

- Warehouse Reception - the destination of all simulated data sets.

Resources:

- Simulation CPU Farm.

4.6 Warehouse Reception

ID: Warehouse Reception

Type: Minor Subsystem

Dependencies: None

Resources:

- Warehouse Reception disk farm.

4.7 Data Warehouse

ID: Data Warehouse

Type: Major Subsystem

Dependencies:

- Warehouse Reception - the source of all data to be stored in this subsystem.
- Data Requisition - the source of queries for data and the destination of any data sets which are the response supplied queries.

Resources:

- Data Warehouse Farm.

4.8 Data Requisition

ID: Data Requisition

Type: Minor Subsystem

Dependencies: None

Resources:

- Date Requisition disk farm.

5.0 Interfaces

5.1 Detector Operations

This subsystem does not present a programmatic interface. However the South Pole Apparatus Management subsystem does need to be able to interact with this subsystem

(using the Experiment Control subordinate of this subsystem). That interface has yet to be defined.

5.2 DAQ Dispatch

The South Pole Apparatus Management subsystem does need to be able to interact with this subsystem. That interface has yet to be defined.

TABLE 1: DAQ Dispatch Interface

Activity	Inputs	Outputs
Gain Access to this subsystem.	(optional) Implementations of subsystem required.	subsystem access object. If the subsystem already exist s an does not match the implementation requested then an Exception is thrown.
Request file for output.	File Metadata. (This contains enough information to make routing and storage decisions without the need to open the file.)	File access object, e.g. file handle.
Release file for output.	File access object. Discard flag (indicates the file should be thrown away, otherwise it will be kept.)	None.
Register interest in files.	Object interested in files (This object will be notified whenever a “kept file” that match the specified selection is released, i.e. available for use.) Selection Criteria.	None.
Request files of interest.	Object interested in files. Selection Criteria.	A list of files access objects that refer to files which the DAQ Dispatch currently contains and that match the specified selection.
Release file from interest	File access object Object interested in files.	None. If the specified interested object has not already expected an interest in the specified file by one of the two proceeding activities then an Exception is thrown.

Some implementation of this subsystem may only present the “output” portion of the interface or the “interest” portion, in which case any invocation on the un-implemented portion will cause and exception to be thrown.

5.3 Data Handling

This subsystem does not present a programmatic interface. However the South Pole Apparatus Management subsystem does need to be able to interact with this subsystem. That interface has yet to be defined.

5.4 Simulation

This subsystem is not a dependency of any other subsystem, therefore it does not present a programmatic interface. However the simulation operator does need to be able to interact with this subsystem. That interface has yet to be defined.

5.5 Warehouse Reception

TABLE 2: Warehouse Reception Interface

Activity	Inputs	Outputs
Gain Access to this subsystem.	(optional) Implementations of subsystem required.	subsystem access object. If the subsystem already exist s an does not match the implementation requested then an Exception is thrown.
Request file for output.	File Metadata. (This contains enough information to make routing and storage decisions without the need to open the file.)	File access object, e.g. file handle.
Release file for output.	File access object. Discard flag (indicates the file should be thrown away, otherwise it will be kept.)	None.
Add file to Warehouse Reception.	File Access object. File Metadata. (This contains enough information to make routing and storage decisions without the need to open the file.)	None
Register interest in files.	Object interested in files (This object will be notified whenever a “kept file” that match the specified selection is released, i.e. available for use.) Selection Criteria.	None.
Request files of interest.	Object interested in files. Selection Criteria.	A list of files access objects that refer to files which the DAQ Dispatch currently contains and that match the specified selection.
Release file from interest	File access object Object interested in files.	None. If the specified interested object has not already expected an interest in the specified file by one of the two preceding activities then an Exception is thrown.

Unlike DAQ Dispatch which is tightly integrated with file production, it is expected that there will be a much more loose couple with Warehouse Reception and its file producers. Therefore the extra “Add file to Warehouse Reception” activity is added to this interface. The expectation is that any file added to the reception this way will be copied into that subsystem.

5.6 Data Warehouse

This subsystem is not a dependency of any other subsystem, therefore it does not present a programmatic interface. However the data manager does need to be able to interact with this subsystem. That interface has yet to be defined.

5.7 Data Requisition

TABLE 3: Data Requisition Interface

Activity	Inputs	Outputs
Gain Access to this subsystem.	(optional) Implementations of subsystem required.	subsystem access object. If the subsystem already exist s an does not match the implementation requested then an Exception is thrown.
Submit requisition.	Requisition. Notification object. (This object will be notified when the requisition have been fulfilled or failed.)	Requisition Identifier.
Delete requisition.	Requisition Identifier.	None.
Register interest in requisitions.	Object interested in requisitions (This object will be notified whenever there is a requisition that matches the specified selection is submitted.) Selection Criteria.	None.
Request requisitions of interest.	Object interested in requisitions. Selection Criteria.	A list of requisition.s that the Data Requisition subsystem currently contains and that match the specified selection.
Release requisition from interest	Requisition Identifier. Object interested in files.	None.
Mark requisition as fulfilled.	Requisition Identifier.	None.
Mark requisition as failed	Requisition Identifier.	None

The system which submits the requisitions as well as the one which fulfills them needs a mechanism for passing details between each other, but which is not directly required by the Data Requision subsystem itself. This is exchange on information is done by subclassing a Requisition object whose interface is as follows.

TABLE 4: Requisition Interface

Activity	Inputs	Outputs
Get Requisition Identifier.	None.	Requisition Identifier.

The rest of this interface has yet to be defined.

6.0 Details

To be completed.