ATLAS SW

Domain decomposition for Atlas software

ATL-SW-Atlas-Requirements-V2 Beta

Contents

Introduction … 1
    Domain dependencies … 1.1

Domains … 2
    Global … 2.1
    Reconstruction … 2.2
    MuonSystem … 2.3
    InnerDetector … 2.4
    Ecal … 2.5
    Database … 2.6
    Event … 2.7
    MagneticField … 2.8
    Geometry … 2.9
    Control … 2.10
    UserInterface … 2.11
    EventDisplay … 2.12
    Unassigned … 2.13
    Detector … 2.14

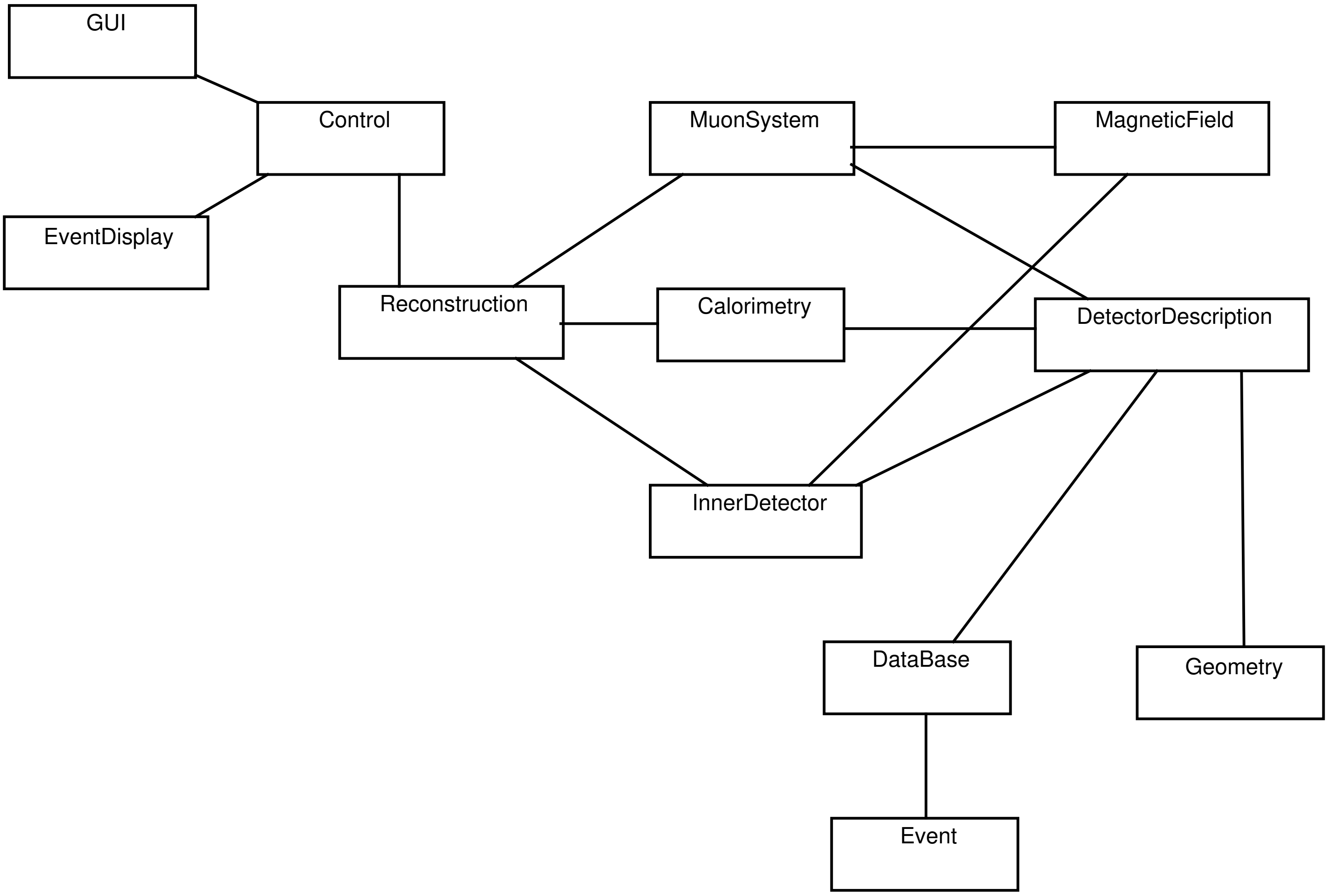
1 Introduction

This document describes the global decomposition of the ATLAS reconstruction and simulation software.

The Reconstruction domain coordinates the work of the detector domains of MuonSystem, Ecal and InnerDetector. They make use of a number of service domains: Control, User Interface, EventDisplay, Geometry, Event and Magnetic Field.

In addition we will need to include domains for, Simulation, DBase and the DetectorDescription, but this has not been done yet.

1.1 Domain dependencies



2 Domains

The requirements which follow are not purely "user requirements" but include "software requirements". They have been grouped by domains and for each domain split into functional requirements, system properties and constraints. It is anticipated that each functional requirement will correspond to real code in that domain. System properties, however, can only be verified after the whole system has been built and do not correspond to code. Constraints impose rules upon the system or upon how it is built.

2.1 Global

In this section we list the properties of the whole system which can only be checked after the system has been built. Any of the domains may also have some required system properties but they should be listed with the appropriate domain. Here only those are mentioned which have to do with the system as a whole such as fault rates and processing speed.

There are numbers mentioned in this section which have been criticised by many reviewers. The numbers are maintained because only those can be verified. It may however be that these numbers will change frequently as our understanding of the system increases.

General constraints upon the system as a whole are listed here. These constraints don't have to correspond to code like requirements do. Many of the constraints listed here are time dependent, with version numbers, and it may be expected that they will change frequently. Unfortunately without explicit version numbers these constraints would be of little value.

2.2 Reconstruction

The major role of the reconstruction domain is to coordinate the activities of the associated domains in order to produce reconstructed objets. This involves the reconstruction, of tracks in the Inner and Muon Detectors, and clusters in the Electromagnetic, Hadronic, and Forward calorimeters. In addition, the reconstruction handles all associations of tracks and/or clusters, originating from the various sub-detectors.

The reconstruction must be able to function with all data types, namely, simulated, test-beam, and real data from the experiment.

An important role of the reconstruction domain is to provide statistical information on global event reconstruction.

The object ROA is used below. A ROA, or region of activity, is a region of the detector which contains data to be reconstructed.This also includes the complete detector. For data from the experiment, a ROA could come from a first or second level trigger. In general, a ROA could be, for example, the region in the inner detector between a significant energy deposition in a calorimeter, and the event vertex. Here, the ROA determines the region in which to look for tracks associated to the calorimeter deposit. In the Inner Detector community, the word, Road, is usually used for a ROA. In the trigger community, Region of Interest, ROI, is also used.

2.3 MuonSystem

The MuonSystem domain is one of the detector domains and is responsible for the reconstruction of tracks in the muon spectrometer. As a second task it might also perform the digitization of hits during simulation.

2.4 InnerDetector

The Inner Detector contains the Transition Radiation Detector, the Silicon Tracker and the Pixel Detector. Each of these detectors produce their own specific type of hits which cannot be combined without any pre-processing. In this pre-processing step neightbouring hits of strips, straws or pixels are combined into clusters and space-points. These steps are specific for the Inner Detector and very different for the Muon Spectrometer and the Calorimeters.

In the current picture this is what belongs to the Inner Detector domain. Maximally one can get a track segment from one subdetector, but the combination of track segments into tracks is part of the Reconstruction domain.

Some of the requirements specified for this domain are similar to those for the MuonSystem. They may evolve to a more general set of requirements for pre-processing. None, or a small number of specific requirements for the individual subdetectors will then remain. For the moment we keep them separate because initially they may correspond to different code. When it becomes clear in the more detailed design which functionality can be shared, the requirements may be re-distributed.

2.5 Ecal

The following requirements apply to the reconstruction of electromagnetic clusters. This covers the first stage of the reconstruction where corrections are applied to the cell energy, clusters are searched for, and the clusters are corrected for calorimeter response variations.

Note that these requirements do not yet correspond to the step of cluster refinement and beyond.

These requirements are valid for both simulated and real data.

2.6 Database

The following are general database functional requirements which are meant to apply to all of the domains which store objects in a database.

Currently the this applies to the event and detector description domains.

2.7 Event

Below are the definitions for the Event domain. Please see the introduction of Event Functional Requirements for a more complete discussion of the Event domain.

The event domain is composed of events and collections of events.

An event is a container for all data which is considered to constitute a physics event throughout the phases of its processing: i.e. raw data collection or simulation, reconstruction, and analysis. This includes the raw data, the reconstructed data, the analysis objects, and the event tag information (these are defined in the Event_Definitions table).

The objects within an event can be classified as either transient, i.e. existing during a processing phase, or persistent, i.e. existing both during and beyond a processing phase.

An event does NOT contain the calibration and alignment constants, nor the reconstruction algorithms used in its processing. These are maintained elsewhere, however, references to the versions of constants and algorithms used in the processing will be contained in the event.

We distinguish between the primary event collection and secondary collections:

The primary event collection refers to the set of events and their organisation when the data if first created, i.e. when the raw data is first put into the database. Subsequent processing steps, e.g. reconstruction, will add information to the primary collection.

Secondary collections will be produced from selected subsets of events, e.g. collections corresponding to physics streams are generated after the first pass reconstruction, and personalised collections will be generated by users during analysis. Secondary collections contain either event references or replicated events.

2.8 MagneticField

The ATLAS detector has a very complicated magnetic field. A super-conducting solenoid surrounds the Inner Detector with an iron return yoke outside the hadron calorimeter. The Muon Spectrometer has 8 super-conducting toroids which will produce a very inhomogeneous field with a lot of field leakage to regions outside the toroids. Because there is also field leakage from the solenoid, almost anywhere in the detector the actual field is a superposition of more than one component.

Note: we still need to formulate a requirement on how to deal with the errors on the field values and tolerances on the trajectory.

2.9 Geometry

Geometry is fundamental to all phases of reconstruction and simulation. In this domain all functionality should be provided to create and work with geometric entities such as points, lines, surfaces and finite volumes. Typical things which need to be created are detector volumes in the simulation and roads for pattern recognition in the reconstruction. Typical operations one may need, are translations and rotations for the simulation to put detector volumes in their proper places in the ATLAS detector. In the reconstruction one often needs to be able to work within the local coordinate system of a subdetector or even of a single chamber but one also needs to able to transform that into the global coordinates of the whole of the ATLAS detector.

A specialised set of requirements came up when a hierarchical description of the ATLAS detector was needed. For that purpose one wants to be able to build a "tree" of detectors where the "root" of the tree is ATLAS in which the Inner Detector, the Calorimeters and the Muon Detectors are put. Inside the Inner Detector are the SCT, the TRT and the Pixel Detector and inside the calorimetrs, the electromagnetic and the hadronic in the barrel and in the endcaps. Etc. etc. This is what is refered to as a "tree" in the following requirements. The detectors in the tree are refered to as "nodes" if they contain other subdetectors, or as "leaves" if they don't. A detector volume which contains other (sub)detector volumes is refered to as a "compound volume". See also the Geometry Definitions.

In the reconstruction one has to deal with measurements which have an error. Therefor the geometry domain has requirements for points and vectors which contain uncertainties.

2.10 Control

The term component is given a very broad definition here in order to explain what is wanted without unnecessarily restricting the implementation.

The control requirements are as far as possible for the control system itself and try to avoid addressing how individual domains make use of the control system.

The control sits between the user interface (or interfaces) and the rest of the system. As such, though important, it is not of direct interest to most users. However the design of the control system could constrain the developers.

We see control in two parts: components and parameters. We assume some kind of component model where components must be interconnected to communicate. One of the tasks of the control system is to establish this communication. This is the users' view of the control system and so should be fast, intuitive and powerful.

The other aspect of control is how a particular object gets its 'control' parameters. We probably want to have a universal interface which gets named parameters from, for example, an Objectivity data base or from a 'data card'.

While it is hoped that the control system will be elegant and flexible, it is also important that it does not introduce too large a time penalty. We set ourselves a quite small target which may be smaller than is necessary considering the ever increasing CPU power available.

The control system is independent of the user interface(s) and may be connected to more than one user interface simultaneously. This would allow, for example, what appears to be to the user a graphical user interface with a menu bar to be implemented as two independent user interfaces, communicating via the control system.

2.11 UserInterface

Different kinds of user interface are envisaged. There may be multiple user interfaces connected to the one control system. It is not anticipated that all types of user interface will have the same level of functionality.

More requirements need to be added for the GUI which address issues such as ease of use, ability to produce screeen dumps etc.

More requirements are also needed for the WEB interface.

Note that nothing is said here about journalling. As we envisage multiple user interfcaes, only the control domain has the complete picture.

Also nothing is said about do/undo, as this is considered to be the concern of the control domain and the controlled objects.

2.12 EventDisplay

The main aim of the Atlas Event Display is to enable visual representation of the objects existing in the Atlas offline software. The Atlas Event Display described in this document should be a part of the whole Atlas offline software package. The basic product should be ready within the year 1997. The development of the first release of the full product is planned for two years.

Event Display interacts both with the data and with the reconstruction package. Event Display consists of a set of views showing geometrical representation of various real objects via graphical objects. Operations are performed on the views and contained graphical objects as well as to original real objects.

Definitions

Real and Graphical Objects: Real objects in the reconstruction program are mapped to their visual representation, into graphical objects of different kinds, corresponding to different views. The graphical objects correspond to the stored real objects and other real objects created from them. Graphical objects can be described as "model" in MVC system.

Views: Different ways of visualising sets of visual objects. View is generally a window with defined way of displaying graphical objects.

Operations: Any actions made in or from the Event Display. Not all operations are applicable to each object or view.

Operations on Grapical Objects: Operations only changing the visual properties of the graphical objects, the real objects are left intact. Visual operations can be performed on the graphical objects or on the whole views.

Operations on Real Objects: Operations on real objects can change the state of these objects, including their visual properties.

2.13 Unassigned

These are requirements which sound useful, but it is not yet clear which domain should have code to satisfy them.

2.14 Detector

Constraints for the ATLAS Detector Description Domain

(Please see Detector_Description_Functional_Requirements for the general introduction of the Detector Description Domain.)

Definitions for the ATLAS Detector Description Domain

(Please see Detector_Description_Functional_Requirements for the general introduction of the Detector Description Domain.)

The ATLAS Detector Description Domain provides a facility to maintain the detector description for the Atlas detector as well as for different test beam activities of the various detector module tests. This database includes the following information:

1) a physical description of the detector, and

2) a logical description of the detector.

The physical description covers the dimensions, shape and material composition of the different types of elements from which the detector is constructed. There is also information which corresponds to each element which is actually manufactured and assembled into a detector, for example the positioning of each element. Both active and passive detector elements must be described. The description of active elements should allow for the specification of deficiencies, e.g. dead channels or broken wires, as well as the inclusion of alignment corrections to the element positioning.

The logical description should provide three primary functions. The first is a simplified access to particular parts of a physical detector description. An example of this would be a hierarchical description where a detector system is described as containing a barrel and two end-caps, each of which is made up of a certain number of layers, etc., and there would be a simple way for a client to use this description to navigate to the information of interest.

The second primary function of the logical description is to provide a means of detector element identification. This should allow for different sets of information which are correlated to specific detector elements to be correctly associated with each other. For example, detector element identifiers could be added to raw data in an event. Then a reconstruction program may build a detector model based on the description in the Detector Description Database and using the corresponding identifiers, be able to associate the raw data of each event with its constructed detector model.

The final function provided by the logical detector description is to support access to subsets of a physical detector description which may be qualified by special "attributes". For example, one may use the attribute "sensitive" to refer to active detector elements. Thus, specifying this attribute for a logical description would allow access to all active elements.

This domain should provide the complete detector geometrical description of the Atlas detector for all of the Atlas online and off-line applications, e.g. the simulation, the reconstruction, the event display, and the online applications.