ATLAS
All trademarks, copyright names and products referred to in this document are acknowledged as such.
ATLAS Collaboration

Armenia
Yerevan Physics Institute, Yerevan

Australia
Research Centre for High Energy Physics, Melbourne University, Melbourne
Australian Nuclear Science and Technology Organisation, Sydney
University of Sydney, Sydney

Austria
Institut für Experimentalphysik der Leopold-Franzens-Universität Innsbruck, Innsbruck

Azerbaijan Republic
Institute of Physics, Azerbaijan Academy of Science, Baku

Republic of Belarus
Institute of Physics of the Academy of Science of Belarus, Minsk

Brazil
Universidade Federal do Rio de Janeiro, COPPE/EE/IF, Rio de Janeiro
Instituto de Fisica, Universidade de Sao Paulo, Sao Paulo

Canada
University of Alberta, Edmonton
Department of Physics, University of British Columbia, Vancouver
University of Carleton/C.R.P.P., Carleton
Group of Particle Physics, University of Montreal, Montreal
Department of Physics, University of Toronto, Toronto
TRIUMF, Vancouver
University of Victoria, Victoria

CERN
European Laboratory for Particle Physics (CERN), Geneva

Czech Republic
Academy of Sciences of the Czech Republic, Institute of Physics, Prague
Charles University, Faculty of Mathematics and Physics, Prague
Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Faculty of Mechanical Engineering, Prague

Denmark
Niels Bohr Institute, University of Copenhagen, Copenhagen

Finland
Helsinki Institute of Physics, Helsinki

France
Laboratoire d’Annecy-le-Vieux de Physique des Particules (LAPP), IN2P3-CNRS, Annecy-le-Vieux
Université Blaise Pascal, IN2P3-CNRS, Clermont-Ferrand
Institut des Sciences Nucléaires de Grenoble, IN2P3-CNRS-Université Joseph Fourier, Grenoble
Centre de Physique des Particules de Marseille, IN2P3-CNRS, Marseille
Laboratoire de l’Accélérateur Linéaire, IN2P3-CNRS, Orsay
LPNHE, Universités de Paris VI et VII, IN2P3-CNRS, Paris
CEA, DSM/DAPNIA, Centre d’Etudes de Saclay, Gif-sur-Yvette

Republic of Georgia
Institute of Physics of the Georgian Academy of Sciences, Tbilisi
Tbilisi State University, Tbilisi
Germany
Physikalisches Institut, Universität Bonn, Bonn
Institut für Physik, Universität Dortmund, Dortmund
Fakultät für Physik, Albert-Ludwigs-Universität, Freiburg
Institut für Hochenergiephysik der Universität Heidelberg, Heidelberg
Institut für Informatik, Friedrich-Schiller-Universität Jena, Jena
Institut für Physik, Johannes-Gutenberg Universität Mainz, Mainz
Lehrstuhl für Informatik V, Universität Mannheim, Mannheim
Sektion Physik, Ludwig-Maximilian-Universität München, München
Max-Planck-Institut für Physik, München
Fachbereich Physik, Universität Siegen, Siegen
Fachbereich Physik, Bergische Universität, Wuppertal

Greece
Athens National Technical University, Athens
Athens University, Athens
Aristotle University of Thessaloniki, Thessaloniki

Israel
Department of Physics, Technion, Haifa
Raymond and Beverly Sackler Faculty of Exact Sciences, School of Physics and Astronomy, Tel-Aviv University, Tel-Aviv
Department of Particle Physics, The Weizmann Institute of Science, Rehovot

Italy
Dipartimento di Fisica dell’ Università della Calabria e I.N.F.N., Cosenza
Laboratori Nazionali di Frascati dell’ I.N.F.N., Frascati
Dipartimento di Fisica dell’ Università di Genova e I.N.F.N., Genova
Dipartimento di Fisica dell’ Università di Lecce e I.N.F.N., Lecce
Dipartimento di Fisica dell’ Università di Milano e I.N.F.N., Milano
Dipartimento di Scienze Fisiche, Università di Napoli ‘Federico II’ e I.N.F.N., Napoli
Dipartimento di Fisica Nucleare e Teorica dell’ Università di Pavia e I.N.F.N., Pavia
Dipartimento di Fisica dell’ Università di Pisa e I.N.F.N., Pisa
Dipartimento di Fisica dell’ Università di Roma ‘La Sapienza’ e I.N.F.N., Roma
Dipartimento di Fisica dell’ Università di Roma ‘Tor Vergata’ e I.N.F.N., Roma
Dipartimento di Fisica dell’ Università di Roma ‘Roma Tre’ e I.N.F.N., Roma
Dipartimento di Fisica dell’ Università di Udine, Gruppo collegato di Udine I.N.F.N. Trieste, Udine

Japan
Department of Information Science, Fukui University, Fukui
Hiroshima Institute of Technology, Hiroshima
Department of Physics, Hiroshima University, Higashi-Hiroshima
KEK, National Laboratory for High Energy Physics, Tsukuba
Department of Physics, Faculty of Science, Kobe University, Kobe
Kyoto University of Education, Kyoto-shi
Naruto University of Education, Naruto-shi
Department of Physics, Faculty of Science, Shinshu University, Matsumoto
International Center for Elementary Particle Physics, University of Tokyo, Tokyo
Physics Department, Tokyo Metropolitan University, Tokyo
Tokyo University of Agriculture and Technology, Department of Applied Physics, Tokyo

Kazakhstan
High-Energy Physics Institute of the Kazakh Academy of Sciences, Almaty

Netherlands
FOM - Institute SAF NIKHEF and University of Amsterdam/NIKHEF, Amsterdam
University of Nijmegen/NIKHEF, Nijmegen
Morocco
Faculté des Sciences Ain Chock, Université Hassan II, Casablanca, and Université Mohamed V, Rabat

Norway
University of Bergen, Bergen
University of Oslo, Oslo

Poland
Henryk Niewodniczanski Institute of Nuclear Physics, Cracow
Faculty of Physics and Nuclear Techniques of the Academy of Mining and Metallurgy, Cracow

Portugal
Laboratorio de Instrumentação e Física Experimental de Partículas (University of Lisboa, University of Coimbra, University Católica-Figueira da Foz and University Nova de Lisboa), Lisbon

Romania
Institute of Atomic Physics, Bucharest

Russia
Institute for Theoretical and Experimental Physics (ITEP), Moscow
P.N. Lebedev Institute of Physics, Moscow
Moscow Engineering and Physics Institute (MEPhI), Moscow
Moscow State University, Institute of Nuclear Physics, Moscow
Budker Institute of Nuclear Physics (BINP), Novosibirsk
Institute for High Energy Physics (IHEP), Protvino
Petersburg Nuclear Physics Institute (PNPI), Gatchina, St. Petersburg

JINR
Joint Institute for Nuclear Research, Dubna

Slovak Republic
Bratislava University, Bratislava, and Institute of Experimental Physics of the Slovak Academy of Sciences, Kosice

Slovenia
Jožef Stefan Institute and Department of Physics, University of Ljubljana, Ljubljana

Spain
Institut de Física d’Altes Energies (IFAE), Universidad Autónoma de Barcelona, Bellaterra, Barcelona
Physics Department, Universidad Autónoma de Madrid, Madrid
Instituto de Física Corpuscular (IFIC), Centro Mixto Universidad de Valencia - CSIC, Burjassot, and Departamento de Informatica y Electronica, Universidad de Valencia, Burjassot, Valencia

Sweden
Fysika institutionen, Lunds universitet, Lund
Royal Institute of Technology (KTH), Stockholm
University of Stockholm, Stockholm
Uppsala University, Department of Radiation Sciences, Uppsala

Switzerland
Laboratory for High Energy Physics, University of Bern, Bern
Section de Physique, Université de Genève, Geneva

Turkey
Department of Physics, Bogaziçi University, Istanbul

United Kingdom
School of Physics and Space Research, The University of Birmingham, Birmingham
Cavendish Laboratory, Cambridge University, Cambridge
Department of Physics and Astronomy, University of Edinburgh, Edinburgh
Department of Physics and Astronomy, University of Glasgow, Glasgow
School of Physics and Chemistry, Lancaster University, Lancaster
Department of Physics, Oliver Lodge Laboratory, University of Liverpool, Liverpool
Department of Physics, Queen Mary and Westfield College, University of London, London
Department of Physics, Royal Holloway and Bedford New College, University of London, Egham
Department of Physics and Astronomy, University College London, London
Department of Physics and Astronomy, University of Manchester, Manchester
Department of Physics, Oxford University, Oxford
Rutherford Appleton Laboratory, Chilton, Didcot
Department of Physics, University of Sheffield, Sheffield

United States of America
State University of New York at Albany, New York
Argonne National Laboratory, Argonne, Illinois
University of Arizona, Tucson, Arizona
Department of Physics, The University of Texas at Arlington, Arlington, Texas
Lawrence Berkeley Laboratory and University of California, Berkeley, California
Department of Physics, Boston University, Boston, Massachusetts
Brandeis University, Department of Physics, Waltham, Massachusetts
Brookhaven National Laboratory (BNL), Upton, New York
University of Chicago, Enrico Fermi Institute, Chicago, Illinois
Nevis Laboratory, Columbia University, Irvington, New York
Department of Physics, Duke University, Durham, North Carolina
Department of Physics, Hampton University, Virginia
Department of Physics, Harvard University, Cambridge, Massachusetts
Indiana University, Bloomington, Indiana
University of California, Irvine, California
Massachusetts Institute of Technology, Department of Physics, Cambridge, Massachusetts
Michigan State University, Department of Physics and Astronomy, East Lansing, Michigan
University of New Mexico, New Mexico Center for Particle Physics, Albuquerque
Physics Department, Norfolk State University, Virginia
Physics Department, Northern Illinois University, DeKalb, Illinois
Department of Physics and Astronomy, University of Oklahoma
Department of Physics, University of Pennsylvania, Philadelphia, Pennsylvania
University of Pittsburgh, Pittsburgh, Pennsylvania
Department of Physics and Astronomy, University of Rochester, Rochester, New York
Institute for Particle Physics, University of California, Santa Cruz, California
Department of Physics, Southern Methodist University, Dallas, Texas
Tufts University, Medford, Massachusetts
High Energy Physics, University of Illinois, Urbana, Illinois
Department of Physics, Department of Mechanical Engineering, University of Washington, Seattle, Washington
Department of Physics, University of Wisconsin, Madison, Wisconsin
# Table Of Contents

ATLAS Collaboration .......................................... iii

List of Figures ........................................... x

List of Tables ........................................... xi

1 Executive summary ....................................... 2

2 Introduction .............................................. 4

3 Computing model .......................................... 6

3.1 Introduction ........................................... 6

3.2 Definitions ........................................... 8

3.3 Requirements ......................................... 10

3.4 Technology trends ..................................... 13

3.4.1 The hardware ..................................... 13

3.4.2 The data management ............................. 16

3.4.3 Conclusions of technology trends ............... 17

3.5 Computing model elements ........................... 18

3.5.1 Standard event reconstruction .................. 18

3.5.1.1 Reconstruction scenario for the initial data-taking .. 19

3.5.1.2 Reprocessing: full and partial ................ 20

3.5.1.3 Generation of the AOD ....................... 21

3.5.2 Monte Carlo production and analysis ........... 21

3.5.3 Physics analysis .................................. 22

3.5.3.1 Access to the event information for analysis ... 23

3.5.3.2 The analysis cycle .......................... 24

3.5.3.3 Sharing of the analysis result ............... 25

3.6 Tools for remote communication and collaboration .. 25

3.6.1 Network infrastructure ......................... 25

3.6.2 Videoconferencing ............................... 26

3.6.2.1 Codec videoconferencing .................. 26

3.6.2.2 Workstation videoconferencing ............. 27

3.6.3 Advanced collaborative tools ................... 28

3.7 Variations of the system architecture ............. 29

3.8 Evaluation ........................................... 30

3.9 Cost estimates ....................................... 32

3.10 Milestones .......................................... 34

3.11 References .......................................... 35

4 Software ................................................ 36

4.1 Requirements for ATLAS software ................. 36

4.1.1 Major requirements .............................. 36

4.1.1.1 Functional requirements .................... 37

4.1.1.2 User control ................................ 37

4.1.1.3 Environmental requirements ................. 37
4.1.2 How the requirements will be realized—need for OO and the ASP  37
4.2 Existing software ............................................. 39
  4.2.1 Calorimeter reconstruction programs .................. 39
    4.2.1.1 Electromagnetic calorimeters ....................... 39
    4.2.1.2 Hadronic calorimeters ................................. 40
  4.2.2 The pattern recognition software for the Inner Detectors .... 41
  4.2.3 The muon reconstruction software ..................... 41
  4.2.4 Simulations of algorithms for LVL1 and LVL2 trigger .......... 44
  4.2.5 Techniques ............................................. 46
    4.2.5.1 ATGEN, an interface to event generators ............ 46
    4.2.5.2 The ATLAS GEometry description language AGE ......... 47
    4.2.5.3 The ATLAS detector description in DICE ............. 47
    4.2.5.4 A C++ interface for the current software .......... 49
4.3 Object-oriented software ...................................... 49
  4.3.1 Traditional model for HEP software .................... 49
  4.3.2 OO model for HEP software ............................... 50
4.4 The ATLAS software process ................................. 51
  4.4.1 Key features of the ASP ................................ 52
  4.4.2 Applying the ASP ....................................... 53
4.5 The ATLAS software development environment .................. 54
  4.5.1 Design methods ......................................... 54
  4.5.2 CASE tool ............................................ 55
  4.5.3 Implementation languages ................................ 55
  4.5.4 Compiler .............................................. 56
  4.5.5 Debugger .............................................. 56
  4.5.6 Programming environment ................................ 56
  4.5.7 Project management .................................... 56
  4.5.8 Verification and validation .............................. 56
  4.5.9 Configuration management ............................... 57
  4.5.10 Documentation ......................................... 57
  4.5.11 Training .............................................. 57
  4.5.12 Communication ......................................... 57
4.6 Strategy for change ........................................... 57
  4.6.1 Benefiting from the existing FORTRAN code base .......... 58
  4.6.2 Building a new software system from scratch ............ 58
  4.6.3 ARVE, an analysis framework design .................... 59
    4.6.3.1 The domains ......................................... 59
  4.6.4 Benefiting from other object-oriented software projects .... 61
    4.6.4.1 GEANT4 (RD44) ....................................... 61
    4.6.4.2 ODBMS (RD45) ....................................... 61
    4.6.4.3 ROOT ............................................. 62
    4.6.4.4 LHC++ ............................................ 62
    4.6.4.5 BaBar ............................................. 62
4.7 Training ..................................................... 62
4.8 Planning .................................................... 64
4.9 Costs ....................................................... 65

viii Table Of Contents
4.9.1 General support tools ........................................ 67
4.10 References ...................................................... 68

5 Trigger and DAQ ..................................................... 70
   5.1 Introduction ................................................... 70
   5.2 T/DAQ Computing .............................................. 70
      5.2.1 Working principles ...................................... 70
      5.2.2 The LVL1 and LVL2 triggers ............................. 72
      5.2.3 The event filter .......................................... 72
      5.2.4 The detector control system ............................ 73
   5.3 T/DAQ software ............................................... 73
      5.3.1 Application software ..................................... 73
      5.3.2 Software environment .................................... 74
      5.3.3 Software development .................................... 75
      5.3.4 Software issues and related R&D ...................... 76
   5.4 References .................................................... 77

A Definitions ....................................................... 78

B Acronyms .......................................................... 80

C Members of the ATLAS Collaboration .......................... 82
List of Figures

Fig. 3-1  p. 10  The logical structure of an event.
Fig. 3-2  p. 18  Global view of the standard event reconstruction.
Fig. 3-3  p. 22  Dataflow diagram for Monte Carlo analyses and their connections to physics and test beam analyses.
Fig. 3-4  p. 24  Dataflow for analysis.
Fig. 4-1  p. 60  Event display, showing detector, a muon track, and response.
Fig. 4-2  p. 63  Training plan (following very much the CERN Technical Training plan).
Fig. 5-1  p. 71  The ATLAS T/DAQ working principles.
List of Tables

Table 3-1  p. 9  The groups of objects for each physics event used in the offline environment.
Table 3-2  p. 13  Estimated evolution of performance and cost for memories, processors, and disks.
Table 3-3  p. 15  Estimate of wide area network costs.
Table 3-4  p. 26  Estimate of network requirements for remote communication and collaboration.
Table 3-5  p. 33  Cost of the ATLAS computing infrastructure.
Table 3-6  p. 34  Spending profile for the central infrastructure.
Table 3-7  p. 35  Milestones for the development of the offline computing system.
1 Executive summary

The effort required to develop and maintain the ATLAS software will be enormous. Because of the dependence of the whole experiment's success and due to the long lifetime of about 20 years of the project, the software quality requirements will have to be very high. For the whole ATLAS software development, up to 1000 person years will be required. It is expected that about 85% of the effort will be from small separated groups not based at CERN. In order to optimize the quality, to guarantee the long term maintenance, and to minimise the necessary resources, a well-defined 'ATLAS software process' is proposed. In the software development we will adhere to accepted international standards; wherever possible we will seek common developments with other experiments and employ commercial solutions. We plan to implement the software following the object-oriented paradigm. Currently, we are studying the implementation using the C++ language.

The data volume produced by the experiment of about 1 Pbyte \((10^{15}\) bytes) per year requires new methods for data reduction, data selection, and data access for physics analysis. The basic consideration is that every physicist in ATLAS must have the best possible access to the data necessary for the analysis, irrespective of his/her location. The proposed scheme consists of archiving the 'raw data' (1 Pbyte/year) selected by the Level-3 trigger system. A first event reconstruction will be performed at CERN, on all data a few hours after data-taking. For this processing, basic calibration and alignment have to be available. The purpose of this processing is to determine physics quantities for use in analysis and to allow event classification according to physics channels. The data produced in the processing have to be accessible at the event level and even below that at the physics object level. We are considering an object-oriented database system for this purpose. One copy of the data will be held at CERN. We also consider replicating some or all of the data at a small number of 'regional centres'. A precise cost estimate for the computing hardware is impossible due to the uncertainties in both the requirements and the evolution of the technology and the market. A rough estimate puts the cost of the central installation of data storage and processing power at CERN for ATLAS to be 20 million Swiss francs, to be spent over several years.

To enable physics analysis in a world-wide collaboration, good networking is a necessity. Today it is impossible to predict the evolution of the cost and the performance of international networks at the time of LHC running. As these are important parameters for the precise planning of an analysis scenario, we have to follow the developments and adjust our planning accordingly. Already during the construction phase, we need international networks for document and code exchange as well as for communication such as videoconferencing in order to minimise travel. Currently, in some areas the networks are still insufficient even for code exchange. CERN plays a central role in many areas:

1. Co-ordination of the software development.
2. Central software and document repository.
3. Support for test-beam data acquisition, data recording, and analysis.
4. Development and maintenance of software packages such as GEANT and the CERN program library.
5. Support of software development tools.
6. Central management of software licences.
7. Support for people located at CERN and visiting scientists in the area of desktop equipment, compute servers, data storage, and networks.
8. Development, installation and operation of computing equipment for the running phase.
While the first three points fall under the direct responsibility of the experimental team, general support on points four to eight should be continued and developed in a central way according to our requirements.

In implementing the computing environment proposed in this document, the critical issues that need to be better understood and solved are:

- Converting a large community to the object-oriented paradigm.
- Implementing a software process including project management for a dispersed developer community.
- Ensuring that the proposed object-oriented database management system for event-data management will provide the required performance in terms of data volume and data access.
- Providing a mass storage system with the required performance.
- Being prepared for changes in the requirements from the experiment and for the evolution in the computing hardware and software environment.
2 Introduction

Computing is of vital importance for the success of the ATLAS experiment. In this document we cover the following elements of computing:

- Software requirements (simulation, calibration, reconstruction, high level event triggering and selection, analysis, visualization).
- Software development process (design and development environment, implementation scheme, quality assurance, organization).
- Computing model (infrastructure requirements in terms of CPU, data storage, networking, analysis model, data processing scheme).
- World-wide collaboration issues.
- Resource requirements (computing infrastructure, software development and maintenance effort, software licences).

These elements are instrumental to the success of the experiment. In many of the areas mentioned above, we do not yet have solutions that are adequate to the scale of the problem in terms of complexity, data rate and data volume, organizational problems due to the world-wide dispersion of resources, and duration of the project. In a rapidly evolving field such as computing, it is impossible and undesirable to fix the detailed design at this stage. Instead, we develop a general strategy and identify areas where further research and development is necessary before a detailed implementation plan can be developed.

In this technical proposal, we can only lay down the requirements as they are known today, technical directions that we currently follow, and a rough estimate of the resources needed.

The organization of computing is part of the overall ATLAS organization. The ATLAS Computing Steering Group (ACOS) deals with offline computing matters and more general computing aspects such as software engineering and computing infrastructure.

The chairperson of ACOS is the ATLAS computing coordinator. He represents offline computing in the ATLAS Executive Board. In ACOS, the computing representatives of the ATLAS systems (Inner Detector, Liquid Argon Calorimeter, Tile Calorimeter, Muon System, Trigger/DAQ) provide direct contact to their respective communities. The detector communities organize their software work relatively autonomously. The coordinators of the major packages such as simulation, reconstruction and trigger simulation integrate the software prepared in these sub-domains. They are members of ACOS as well as the software librarian, the resource coordinator, the software engineering coordinator, the chairman of the computing model group, and the CN contact person. Additional members represent specific geographical regions within the ATLAS collaboration.

In the document we cover the detector design and optimization phase, the construction and commissioning phase and the exploitation phase.
3 Computing model

3.1 Introduction

The ATLAS computing model describes the global architecture of how we plan to use software, processing power, storage and networks to do the offline computing at LHC. The term offline computing encompasses detector calibration and alignment, event reconstruction, Monte Carlo generation, and physics analysis of both real and simulated data. The basic inputs to the model are:

- 100 Hz event rate out of the Level-3 trigger, i.e. $10^9$ events per year;
- 1 Mbyte event size;
- ~1 Mbyte/h of calibration and alignment data,

leading to ~1 Pbyte (10$^{15}$ bytes) of raw data per year. Our model must take into account both low-luminosity running (10$^{33}$ cm$^{-2}$s$^{-1}$), expected at beam turn-on, as well as high luminosity running (10$^{34}$ cm$^{-2}$s$^{-1}$). It is assumed that the above data rate remains essentially constant as the luminosity increases by a factor of 10.

The event reconstruction must handle the 100 Hz rate out of the Level-3 trigger. We propose to reconstruct quasi-online allowing for an ~1 hour delay for the generation of the alignment and calibration constants from the data themselves. For the output of the reconstruction, we target an event size of ~100 kbyte, i.e. a reduction of a factor of 10 in data volume. The set of objects produced by the reconstruction is labelled Event Summary Data (ESD). It is anticipated that the reprocessing of events, to account for changes in the calibration, alignment, or reconstruction algorithms, can begin from either the ESD or the raw data. We propose to allocate sufficient resources to reprocess events: a few times per year starting from the ESD, and once per year starting from the raw data.

There are five activities requiring access to the data which are of interest for the computing model:

- Monitoring the detector performance and the data quality.
- Understanding the detector response: calibration, alignment.
- Developing and testing of reconstruction algorithms.
- Studying high-$p_T$ physics.
- Studying B physics.

The monitoring will run in parallel with the event reconstruction to provide rapid feedback during data-taking. The detector response studies need access to a variety of events from calibration to specific physics channels and, as well, need access to both raw and reconstructed data. Similarly, the development and testing of reconstruction algorithms will to a certain extent be done with Monte Carlo events before the beginning of data-taking. However, when confronted with real data these algorithms will require tuning, and there will be further developments as new ideas arise.

---

1. These are assumed to have an uncertainty of a factor of 2.
2. Trigger cuts and event size will be adjusted to maintain an approximately constant data rate.
For the physics studies, the computing model must allow efficient access to select and study relatively small event samples embedded in large samples of mostly background. For example, many physics channels consist of $10^7$ to $10^9$ events, i.e. 1 - 10% of the annual event sample. One can imagine that several groups apply different selection criteria to define `analysis samples' which are several orders of magnitude smaller. These are then extensively studied, resulting in a new set of selection criteria which is used to repeat the exercise.

The information required for physics studies is generally just simple `physics objects', i.e. electrons, muons, jets, tracks, etc., requiring only a small amount of data per event (relative to the initial 1 Mbyte), estimated to be less than $\approx 10$ kbyte. This set of objects is labelled Analysis Object Data (AOD). An important point for the computing model is that these `physics objects' evolve with time as the calibration constants and reconstruction algorithms improve. This is particularly true during the early phase of the experiment where the first physics data will be used to understand the detector response.

Monte Carlo studies have already been extensively exploited for the design of the detector and trigger and the understanding of test-beam results. Studies will continue during the construction phase, and as well, Monte Carlo generated events will be used to test the offline event reconstruction and analysis software. As the experiment begins taking data, the Monte Carlo will need to be tuned and then used to calculate corrections for physics results. It is estimated that the required number of Monte Carlo generated events is approximately 10% of the number of real events, and corresponds to $\approx 5 \times 10^4$ SPECint95\(^1\) processing power. The low I/O bandwidth required for Monte Carlo generation allows it to be distributed across the collaboration. This effort will need to be organized collaboration-wide, and most likely only the ESD and/or AOD information will need to be made available for general use.

Technology is an important ingredient in the computing model, since the offline system which will eventually be designed relies on the capabilities of an underlying technological layer. The extrapolation of cost estimates for networks, storage and computing power is difficult over a time-scale of several years. However, for the requirements of the computing model presented in this chapter, it is reasonable to expect from recent trends that the cost of storage and computing power will have decreased sufficiently for the requirements of ATLAS to be satisfied. The largest uncertainty lies in the affordability of wide area network (WAN) bandwidth, in particular because of the deregulation of the European telecommunications industry and the recent rapid growth in Internet usage. The importance of WAN bandwidth becomes clear when one understands that, in order to analyse the large volumes of data produced at LHC, the processing power is required to be close to the data, and thus analysis facilities will be localized at CERN and possibly a few regional centres.

Software is also an important technological element in the computing model. Most of the software development issues for ATLAS are treated in Chapter 4. However, the key software elements which directly concern the computing model are the management and the storage of data. Here there are two components of importance:

- Object Database Management System (ODBMS)
- Mass Storage System (MSS).

Commercial ODBMS and MSS capabilities are currently under study by RD45 [3-1]; their preliminary results are promising. An ODBMS would serve as a front-end tool where one organizes and manages the data from a logical perspective, i.e. one directly manipulates runs of events, individual events, tracks of an event, different samples of events, etc., and the ODBMS manages

---

1. The definition of SPECint95 can be found on page 10.
the physical location of the information, i.e. which part of an event is stored on which file. An MSS would serve as a large bandwidth back-end file server allowing hierarchical storage management1 of the data which is transparent to the front-end user. The current view is that the combination of a commercial ODBMS and MSS will manage all of the data for both the event reconstruction and the physics analysis.

We expect that there will be \( \sim 500 \) 'equivalent physicists' performing some analysis task with \( \sim 150 \) users simultaneously accessing the data. We assume that from start-up all physicists will have adequate access to the data to perform analysis from their home institute. It will be important that there is a coherent view of the data independently of where the data physically resides, i.e. at CERN, at a regional centre or at one's home institute.

The question of the rôle of regional centres in the ATLAS computing model has not yet been resolved. It is generally agreed to perform the event reconstruction at CERN. Also, the bulk of the raw data will remain at CERN. Thus, any reprocessing of a large fraction of the raw data will be done at CERN. The rôle of regional centres would be to concentrate on the areas of physics analysis, MC generation, and possibly some of the reprocessing which begins from the ESD information. The information provided in the following sections is intended to begin the preparation for a decision that will be taken by the end of 1998.

The participating institutes in ATLAS provide the basic support for their physicists. This includes desktop support and a certain amount of computing power and storage. The rôle of the institute within the ATLAS computing model will also need to be understood. The key point is to provide the resources so that one can perform the required analysis tasks from the home institute. This may include some data which is physically transferred. However, it should be stressed again that the majority of the data will have to remain at the large facilities, i.e. CERN and possibly regional centres.

In the following sections we present in detail the ATLAS computing model. A few definitions are given in Section 3.2. An overview of the computing model requirements are discussed in Section 3.3. A survey of technology trends is given in Section 3.4. A description of the different elements of the computing model is given in Section 3.5. A separate discussion on the tools for communication and collaboration required both during the construction and the running phase of the experiment is given in Section 3.6. Variations of the system architecture are given in Section 3.7 and their evaluation and conclusions are given in Section 3.8. Cost estimates for the offline system can be found in Section 3.9. Finally, the various milestones for the development of the offline system and the corresponding time-scales for decision taking are discussed in Section 3.10.

### 3.2 Definitions

We introduce here a few terms which will be used throughout this chapter.

One regards all of the experimental data, i.e. the event data, calibration and alignment data, user analysis data, etc., as residing in a single federated database:

**Federated database:**
This refers to a set of individual databases which are managed coherently. Each database contains a logically connected group of objects, e.g. the raw data for events of a single run. Objects

---

1. Hierarchical storage management refers to migrating files between, for example, fast random access disks and slower sequential access tapes according to the access demand.
can hold references to each other, allowing navigation, even from one database to another. A
database represents a single file and objects are stored on database pages. When access to an object
is required, it is one or more database pages which are transferred from storage to a memory
cache. A key to performance is to minimize the disk/memory transfers which can be optimized
by clustering objects which are likely to be used together onto the same database page. A federated
database is a distributed database where the individual databases can be distributed across
a number of nodes and/or sites.¹

**Event object groups:**
The various objects of an event are grouped together to define different event object groups
used in the offline analysis: raw data, ESD, AOD and event tag. A given object might be part of
one or more event object group(s). These definitions and an estimation of their size are given in
Table 3-1.

<table>
<thead>
<tr>
<th>Event object group</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw data</td>
<td>1 Mbyte</td>
<td>information coming out of Level-3</td>
</tr>
<tr>
<td>Event summary data (ESD)</td>
<td>100 kbyte</td>
<td>reconstruction information in enough detail to do event display, generate analysis objects, and redo most of the reconstruction.</td>
</tr>
<tr>
<td>Analysis object data (AOD)</td>
<td>&lt;10 kbyte</td>
<td>physics objects, e.g. electrons, muons, etc., used for analysis</td>
</tr>
<tr>
<td>Event tag</td>
<td>&lt;100 bytes</td>
<td>brief information allowing a rapid first-pass selection to find events of interest</td>
</tr>
</tbody>
</table>

It is useful to distinguish between the *data definition* and the *data content* of the event objects:

**Data definition (or schema definition):**
this traditionally has been referred to as the ‘data (or bank) format’, and in an ODBMS world
would be done using an object definition language (ODL) which has a syntax similar to C++ to
specify classes and relationships.

**Data content:**
this corresponds to the values stored for each object. Note that the data content depends on the
calibration, alignment and reconstruction algorithms, as well as the criteria which define each
physics object, e.g. the isolation criteria used to identify an isolated electron.

¹ *Federated database* is the term used by Objectivity/DB to describe its architecture. Objectivity/DB is the
commercial system currently favoured by RD45 (see Section 3.4.2). Strictly speaking, the correct term to
describe this architecture would be simply a *distributed database*, and a *federated database* should refer to
multiple databases from different vendors which are connected together via *gateways*. For a complete
description of the Objectivity architecture, one would have to add that objects are stored in *containers*
which in turn are stored in a database. The clustering of objects is done at the container level. The logical
to physical mapping is: a database ⇔ one disk file, and a container ⇔ one or more database pages.
From the perspective of the federated database, one can view these parts as being logically connected together to form a single event, as is shown schematically in Figure 3-1. The lines connecting the different parts imply that one can physically navigate from part to part. For example, one may loop over the event-tag objects of a set of events and use event-tag information to define a subset for further study. One is then able to loop over this subset and access the ESD information of each event by navigating through an ‘event_header’ object. It should be stressed that this logical event structure implies nothing about the physical location of the different parts. For example, one could have the event-tag information stored in a database which is cached onto a physicist’s workstation in his/her local institute and the rest of the event stored in other databases at CERN.

Database collection:
a container which holds either references to objects or the objects themselves. The objects held in a collection can always be accessed, sequentially or otherwise, using another object called an iterator. In terms of the previous example, the set and subset of event-tag objects would be realized as database collections which are iterated over.

Regional centre:
a concentration of storage and processing power to be used primarily for analysis and MC generation. A regional centre would be part of the ATLAS federated database and would contain replicated data from the main CERN site. The physical location is not specified, meaning that one or more regional centres may be located at CERN.

SPECint95:
This is a processor benchmark which is used in this document as the unit of processing power. A SPECint benchmark is most applicable to HEP code because the code is dominated by integer manipulations. For comparison, 1 SPECint95 = 40 SPECint92 = 10 CERN units = 40 MIPS. To put this into perspective, the ATLAS work group server HP/D250 machines introduced in autumn 1996 have PA7200 processors at 100 MHz rated at 3.6 SPECint95, and the SHIFT HP/K400 machines have PA8000 processors at 180 MHz rated at 10.4 SPECint95.

3.3 Requirements

Alignment and calibration constants of sufficient quality for first-pass processing must be available within a few hours after data collection.

Raw data of 1 Mbyte per event must be recorded at a peak rate of 100 Hz.

1. See footnote 5 on page 16 for an explanation of replication.
A fast-access buffer must be provided to store the raw data between the output of the Level-3 trigger and the first-pass processing.

A data archive system must be provided to store 1 Pbyte of raw data per year.

Raw data need to be available and accessible

- for detector verification studies (access to a small fraction of the data, infrequently);
- for calibration purposes (access to a fraction of data, very infrequently);
- for alignment purposes (access to selected data, infrequently);
- for detailed event visualisation (access to a very small fraction of data);
- for a first pass processing which reconstructs the basic event parameters, performs event classification, and produces the ESD (access to all the data, once);
- for reprocessing of event parameters (either accessing all data once, or selected data more frequently).

While most of the raw data can in principle be restricted to physically remain at the central CERN location, they should be easily accessible from anywhere in the collaboration.

Physics quantities such as four vectors of identified particles, jet parameters, energy flow, missing energy, and short-lived decays need to be calculated at the reconstruction stage with the best possible accuracy.

ESD will constitute, for the initial period of the experiment, the basis of any more detailed processing and of all physics analysis. They need to be accessible from all ATLAS institutions participating in physics analysis.

Virtual samples\(^1\) of events which have been classified according to the physics analysis channel need to be accessible.

AOD must provide optimized access to the data most often used in analyses.

Event-tag objects

- must allow flexible event classification according to the physics analysis needs
- must be accessible from all sites with best possible speed.

Navigation between raw data, event summary data, analysis object data, and event tag data must be provided in an optimized way.

A partial (in terms of selected events or selected objects) update of the objects belonging to event summary data, analysis object data, or event tag data must be possible.

The status of the data of each event\(^2\) must be precisely tagged in terms of algorithm version, as well as calibration and alignment data version, in order to provide a unique and repeatable recognition of data.

--

1. Virtual samples refers to collections of event references, i.e. events that fall into different samples are not copied into each corresponding collection, rather each collection contains a reference to the same event.
2. Each atomic part of an event, i.e. each object or group of objects which can be updated separately, must necessarily have its own status tag.
Fast access (e.g. on a direct-access medium) must be provided for the 100 Tbyte of data per year which contain the event summary data, the analysis object data and the event tag.

Personnel to operate the data processing and the data storage systems with a reliability of >99% is required (less than four days of unavailability per year).

An hierarchical storage management system to allow flexible allocation of the direct-access media is required.

Up to 150 physicists must be able to access the database and perform analyses simultaneously.

Networking between a central data store and all institutions must be provided to

- transfer the raw data, event summary data or just the graphics objects necessary for event display at remote locations;
- transfer the result of a data query primarily in the form of histograms, and some reduced amount of selected event summary data, analysis object data or Ntuples.

The minimum allocated network bandwidth required for analysis between each user and the analysis centre used must be 1 Mbit/s.

Either the minimum allocated bandwidth between CERN and each regional centre must be 480 Mbps for the transfer of all ESD twice per year by network, or there must be a corresponding facility for a total transfer to each regional centre of 200 Tbyte of data by movable media (e.g. magnetic tapes or optical disks).

In the collaboration, sufficient capacity (in terms of processing power and of data storage) is required to produce the simulated data necessary for

- detector design;
- detector calibration;
- physics analysis.

Sufficient processing power must be provided for

- calibration and alignment necessary for first-pass event reconstruction;
- first-pass event reconstruction generating the ESD;
- second-pass event reconstruction for all events starting from the raw data (once per year);
- redoing a partial event reconstruction from the ESD itself (up to twice per year);
- generating analysis object data (several times per year);
- data access for analysis of all physicists in the collaboration;
- physics analysis for all ATLAS physicists, performed at CERN, at a Regional Centre, and to a lesser degree at the home institute.

---

1. It is understood that due to the large volume of data most of the data during analysis will remain in the central data store(s).

2. Allocated bandwidth is, today, the bandwidth for which one pays. This corresponds on average to a delivered bandwidth of ~1/10 of the allocated bandwidth for typical interactive analysis work and to ~1/3 for bulk data transfer as given in the next requirement.
3.4 Technology trends

The technological capabilities at the time of LHC start-up are certainly one of the more important constraints for the computing model. Since the data volume rates in ATLAS will be two orders of magnitude higher than in current collider experiments, e.g. pp Fermilab or HERA, satisfying the requirements will depend on the continued systematic increase in computing capabilities which has been the norm in the computing industry. We summarize here the current predictions for the evolution of the technology which are relevant to the computing model. It must be kept in mind that it is not possible to predict with great accuracy the state of technology in 2005, and so for planning purposes periodic updates will be necessary. The relevant technological elements are the hardware (memory, processing power, disk and tape storage, and networking) and the data management (object databases and mass storage systems). The following has been extracted from references [3-2], [3-3], [3-4].

3.4.1 The hardware

Table 3-2 presents the current and the 2005 projected characteristics for memory and processors, taken from [3-2].

<table>
<thead>
<tr>
<th></th>
<th>1996</th>
<th>2005</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Memory</strong></td>
<td>0.35 mm feature size</td>
<td>0.13 mm feature size</td>
</tr>
<tr>
<td></td>
<td>64 Mbits/chip</td>
<td>4000 Mbits/chip</td>
</tr>
<tr>
<td></td>
<td>30 CHF/Mbyte</td>
<td>50 CHF/Mbyte</td>
</tr>
<tr>
<td><strong>High-end processors</strong></td>
<td>10 SPECint95a</td>
<td>100 SPECint95</td>
</tr>
<tr>
<td></td>
<td>25 kCHF/system</td>
<td>5 kCHF/system</td>
</tr>
<tr>
<td><strong>Low-end processors</strong></td>
<td>5 SPECint95</td>
<td>40 SPECint95</td>
</tr>
<tr>
<td></td>
<td>6 kCHF/system</td>
<td>1 kCHF/system</td>
</tr>
<tr>
<td><strong>Magnetic disk</strong></td>
<td>400 CHF/Gbyte</td>
<td>12 CHF/Gbyte</td>
</tr>
<tr>
<td></td>
<td>6 Mbyte/s-diskb</td>
<td>multi-Gbyte/s per system</td>
</tr>
</tbody>
</table>

a. For a definition of SPECint95 see page 10.
b. Today disks can be striped or combined in RAID (Redundant Array of Inexpensive Disks) to provide parallel access reaching ~100 Mbyte/s.

Memory is expected to quadruple in capacity every three years due to decreasing feature size and increasing chip size. The fabrication facilities are currently financed and/or built for a feature size of 0.18 μm, and thus, new technology will be needed to reach the 2005 numbers. Memory capacity is not a major issue for LHC event processing farms - default system memory will be sufficient for event-by-event processing. However, the dropping cost will certainly influence the interactive analysis facilities where 10-100 Gbyte memory will be affordable. Here one can begin to imagine systems where frequently accessed data are maintained in memory, allowing to effectively utilize memory access speeds which are over 1000 times faster than disk access.

From Table 3-2 one sees an estimated increase of 10 in unit computing power and a factor of 50 in computing power per unit cost. Note that this estimate includes a speculative increase of a factor of three from the number of instructions executed during each chip cycle. It is not clear

1. Costs have been estimated in $US and converted to CHF with an exchange rate of 1.25 CHF/$.
today whether this type of parallelism will be pursued by the manufacturers. From these estimates, the very large aggregate processing capacity required for ATLAS appears to be affordable, but this will require an enormous number of processors. Thus, how these processors are interfaced, or the system architecture, becomes an important issue. One potential architecture is Scalable Parallel Processors where physically independent computers are interconnected by a high performance switch with a special interface which enables it to be used with a low hardware and software overhead (as opposed to a standard network connection). See [3-2] for further details.

The price/capacity of random access magnetic disks is currently improving at a rate of 30-40% per year, which if sustained, will result in a cost of about 12 CHF/Gbyte for a packaged disk in 2005. It should be noted that disk capacity and cost are driven by the large demand of the PC market. It is this demand that will determine the basic disk characteristics, e.g. access time, transfer rate. However, these are not sensitive factors for LHC since bandwidth needs can easily be satisfied with parallel I/O using software striping and RAID disk subsystems. Thus one can safely predict that multi-100 Tbyte disk farms will be possible at LHC start-up.

Traditionally HEP has used magnetic tape technology to store the experimental data, and disks are used for copies of a ‘frequently accessed’ fraction. Because market pressures are different for disk and tape, it is possible that roles may change in the future. The demand for magnetic tape technology is driven by the market for backup and data archive. This market requires storage for data which is written and rarely read (whereas we write once and read often), and has a growth rate which is coupled to the disk storage capacity. Overall, it is not so much a question of technological development but of market demand that will determine the tape capacity and read/write speed. Currently, tape speed is 10 Mbyte/s and tapes can contain ~50 Gbyte or more, depending upon the technology, and the costs are 20-60 CHF per robot slot and 125 CHF per tape, giving an automated storage cost of 2.75-3.15 CHF/Gbyte. It is difficult to extrapolate these numbers since there is currently little market pressure for either tape speed or capacity. Conservative estimates for the year 2000 would be 50 Mbyte/s, 100 Gbyte per tape and a total automated storage cost of 0.50-1.0 CHF/Gbyte. Further extrapolation is not possible. There may be possible alternatives to magnetic tape, e.g. digital video disk, optical tape, recordable CDs, and holographic storage. However, a serious alternative would have to be well established by the year 2000 to be able to be chosen as the tertiary storage medium for LHC.

Networking requirements at LHC can be divided into three categories:

- Interactive analysis: for example for those who are connected to the federated database launching jobs and examining histogram results.
- ‘Background’ interactive work: for example file transfers, computer supported cooperative work, and videoconferencing/multicast services.
- Bulk data transfers.

For on-site services, there is no foreseeable problem in satisfying the demand. A point-to-point connection in the 2 to 5 Gbit/s range will be available to transmit the 100 Mbyte/s data rate from the experimental area to the central recording location. One can also expect an on-site interactive bandwidth of 100 Mbit/s to the desktop.
The evolution of the cost of off-site connections, or wide area networks (WAN), is highly uncertain. The tariff structure of network services provided by the European Telecom Companies (Public Network Operators) in certain Member States is currently still well above the cost, and it is unclear that the full liberalisation of the European Telecom market starting 1st January 1998 will give way to effective competition and cost-oriented pricing\(^1\). Table 3-3 presents the current costs of WAN and gives an estimate for the costs in 2005. The lower number for the extrapolated cost is considered 'optimistic' and there is a factor of 5 for uncertainty.

**Table 3-3** Estimate of wide area network costs.

<table>
<thead>
<tr>
<th>Point-to-point connection</th>
<th>1996</th>
<th>2005</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Europe(^a)</td>
<td>Intercontinental</td>
</tr>
<tr>
<td><strong>Lower speed</strong></td>
<td>kCHF/year</td>
<td>kCHF/year</td>
</tr>
<tr>
<td>Typical cost of 1 Mbps circuit</td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td><strong>Higher speed</strong></td>
<td>MCHF/year</td>
<td>MCHF/year</td>
</tr>
<tr>
<td>Typical cost of 155 Mbps circuit</td>
<td></td>
<td>0.6-3</td>
</tr>
<tr>
<td>Typical cost of 622 Mbps circuit</td>
<td></td>
<td>1.4-7</td>
</tr>
<tr>
<td>Typical cost of 2.4 Gbps circuit</td>
<td></td>
<td>2.5-12.5</td>
</tr>
</tbody>
</table>

\(^a\) Averaged cost of domestic and international circuits.

Today the cost of national leased lines is about 50-70% lower than for international ones. By 2005, this difference should be reduced to 10-20%. As well, today there is a difference of up to a factor of 5 in the domestic rates between Europe and the USA. It is estimated that in 2005 the European rates will be much closer to those in the USA, with the USA retaining perhaps a 30-50% advantage. Finally, when considering networking costs in a computing model, one should keep in mind that in some cases domestic network bandwidth costs can be hidden by internal infrastructure whereas more dedicated international connections may not.

A new international networking standard, Asynchronous Transfer Mode (ATM), is beginning to be deployed. ATM is a transmission, switching and signalling technique which can be used for both WAN and LAN, operating at a wide range of speeds from Mbit/s to Gbit/s. ATM can accommodate mixed media traffic types (data, voice, images, video) and offers a *quality of service* which can assure a certain bandwidth for an application ('bandwidth-on-demand'). ATM is the foundation technology for broadband ISDN. ATM has better performance because it provides connectivity through a switch as opposed to, for example, an Ethernet shared bus. However, the transition to ATM is not so obvious, because Ethernet is well established and still improving; it is now possible to have fast Ethernet switches connecting 100 Mbit/s shared segments, and desktop boxes come with default Ethernet connections. So it may be that ATM will be widely used by telecommunications operators within their networks, but will only penetrate LANs as a backbone with a fast switched-Ethernet, or equivalent, layer on top.

---

1. Analysts have voiced doubts that even 'competition' will have the expected benefits on prices, at least during the first decade after liberalisation. There is a tradition, and many examples of 'agreements' between suppliers of telecom services.
Internet Protocol (IP) will probably remain the main communications technique over the next 15 years. IP routers will persist in both LAN and WAN, evolving towards faster switching fabric. In running IP over ATM, it will be difficult to exploit the quality of service guarantees of ATM at the application level. However, quality of service guarantees will be possible only for community Internet, as opposed to public Internet, where all the routers are under the control of a community. The current trend for HEP is a move away from a HEP-specific leased lines to more general scientific networks. In order to assure the bandwidth eventually required for the specific activities and tools discussed in detail in Section 3.6, ATLAS needs to carefully study the deployment and costs of additional, dedicated network connections.

3.4.2 The data management

With the adoption of the object paradigm, one must find a solution for managing the input/output of the data that will be produced at LHC. Traditional techniques, e.g. file-based ZEBRA I/O, do not map well onto the object model, nor do they scale to the requirements at LHC. An interesting possibility, currently being investigated by RD45 [3-5], is Object Database Management Systems. This is a powerful solution capable of scaling to handle the multi-Pbyte of data expected in a fully-distributed, heterogeneous environment. A federated database architecture provides a single logical view of a whole data store, e.g. all of a single experiment’s data. An ODBMS integrates I/O with the programming language, e.g. C++, and makes I/O transparent to the user/application. Furthermore, adhering to standards when using an ODBMS provides both language and vendor independence in that objects stored in one language, e.g. C++, may be retrieved with another, e.g. Java. This provides transparent access to any object in the federated database; for example, knowing the run number, event number and the subset of event information desired, one can navigate to this information without knowing either the file, the directory, or even the physical location of the file system. There are also other database features which can improve data management and the performance of the system. For example, replication of user data in a distributed environment can improve access performance, availability of data, and autonomy for those working in different sites/network segments. Another important feature is schema evolution where there is support not only for adding new classes (data types) to the database, but more importantly for dealing with the impact of changes to class definitions on the data already in the database.

RD45 is now in its second year of investigation. The members of this project have shown, on paper, that a system designed around an ODBMS is a potential solution, significantly more power-

---

1. When setting up an ATM connection, the sender specifies the type, speed and other attributes of the call, which determine the end-to-end quality of service. Different types of service can be, for example, for data (which tends to be large quantities of information needed as fast as possible) or for voice and video (which tend to be more even in the amount of information required, but are very sensitive to when and in what order the information arrives).
2. Present day experiments have already hit limits with ZEBRA in treating Tbyte of data.
3. A federated database is defined on page 8.
4. Language independence is possible when one defines classes using the Object Definition Language (ODL), a de-facto standard defined by the Object Database Management Group (ODMG), which was established by a group of database vendors. Currently it is defined how ODL is connected to C++ and Smalltalk, and the binding to Java will occur in about 1998. More importantly, adherence to the ODL standard also permits an application to a large degree to be vendor-independent, allowing an application to be ported from one ODMG compliant vendor to another.
5. This is when a set of objects are physically duplicated, typically on separate database servers in different locations. The system maintains the coherence of the different copies, e.g. changes to one copy will be propagated automatically to the other copies.
ful than today's file-based solution, that can scale to the requirements of LHC. They have identified two commercial products (Objectivity/DB and Versant) which can potentially scale to our needs, and they are working closely with one of the commercial suppliers, Objectivity, to understand one of the key issues: performance. An ODBMS solution would have all data (raw data, Event Summary Data, and Analysis Object Data) in the same federated database, hence supporting both event reconstruction and analysis. Thus, one views the data as being stored in a single logical database, even though it would be distributed over multiple servers managing multiple subsets. Currently, one of RD45's key milestones is the evaluation of the effectiveness of an ODBMS as a query and access method for physics analysis. This evaluation, to be performed with a federated database of ~100 Gbyte, will indicate whether the ODBMS is sufficient to obviate the need for the use of Ntuples and allow the added flexibility of having access, when needed, to more detailed information of an event.1

In addition to an ODBMS, there is need for a Mass Storage System (MSS) which essentially renders transparent whether files are stored on a rapid-access medium, e.g. magnetic disk, or on a 'backup' medium, e.g. tape. Although the fraction of data that will be kept on disk in 2005 cannot be reliably predicted, one must conservatively estimate that not all data will be on disk and an MSS will be required. The mass storage market has recently been given a significant boost with IBM's announcement, in September 1996, of the availability of new High Performance Storage System software (HPSS). This has been developed through a collaborative effort between industry, government and research sites. HPSS offers a scalable solution which can eventually attain the required capacity of multiple Gbyte with throughput per transaction in the multiple Gbyte/s range. There is currently an ongoing effort to integrate Objectivity's DBMS with HPSS. This integration will be done in such a way as to render transparent to the application whether a database is on tape or disk. Thus, a combined ODBMS and MSS system will be tested by RD45 in 1997.

Finally, there is the question of file systems - the basic container offered by operating systems. UNIX is currently moving to 64-bit file systems which will allow to have database files in the required tens of Gbyte range.

3.4.3 Conclusions of technology trends

From the basic requirements of ~1 Pbyte/year of raw data, ~100 Tbyte/year of ESD and ~10 Tbyte/year of AOD for analysis, one can reasonably expect that with a reasonable budget the necessary hardware to construct the event reconstruction and analysis systems will be affordable. This, of course, assumes that the current trends of improving price/performance are maintained. The key element which will remain uncertain for some time to come is the level of affordable networking.

Concerning data management, we consider the potential of an ODBMS to be of considerable interest, and support RD45 in their investigations. We consider that it is reasonable that a commercial ODBMS with the capabilities described in [3-4] will be available well before 2005, and thus consider an ODBMS as ATLAS' baseline option for data management. It is clear, however,

1. Ntuples are currently used to have fast access to a small amount of information needed for a particular analysis. With an ODBMS, one accesses directly individual objects and thus a separate system supporting Ntuples may no longer be necessary. Reclustering parts of an event may be required for performance, and one will need facilities for user-defined objects as one has with Ntuples. But the hope is that one can obtain the required performance while remaining within the database and thus maintain access to the rest of the data.
that it has not yet been demonstrated that an ODBMS can satisfy all of the requirements of ATLAS, and we await further milestone results from RD45.

3.5 Computing model elements

3.5.1 Standard event reconstruction

The standard event reconstruction accepts the events coming from Level-3, generates the necessary calibration and alignment constants from the events themselves and performs the reconstruction needed to generate the ESD information. It also generates the AOD objects from the ESD. The events are flagged\(^1\) to form logical streams, or events sets, for the different physics channels or other special categories of events. This can be viewed schematically in Figure 3-2.

![Diagram of event reconstruction process]

Figure 3-2 Global view of the standard event reconstruction.

It is assumed that the formats of the raw data and ESD are defined uniquely for the whole of ATLAS. This, of course, does not exclude the evolution of the definitions. From past experience, e.g. CDF, it has been found to be advantageous to define also a standard AOD format for all physics working groups. These common definitions correspond to both the data definition and the criteria used to define each object, e.g. what is an EM cluster or a muon. This will help people to use the different event samples. All of the information which defines the different formats

---

1. Flagged means that access information is provided to study only a subset of events with given properties. From the database perspective, any single event can appear as if it belongs to two or more physics streams without physically being copied. A subset of events would form a database collection of event references.
of an event (schema, selections, criteria) will most likely be stored in the ODBMS and versioned, and have associations to the corresponding event collections.

We assume that all of the data will reside in a single federated database. However, it is too early to decide how the different event-object groups will be stored, for example how to divide the objects into separate databases and how to cluster them. This will be the subject of future studies.

Since the standard event reconstruction is a well-defined task and the volume of raw data is so large, we find it mandatory to concentrate the event reconstruction in one place, the natural choice being CERN. We assume as well that the bulk of the raw data will remain at CERN and that all the subsequent reprocessing which starts from the raw data will also be done at CERN. This, of course, does not exclude the ‘export’ of small amounts of raw data to other sites as needs arise. For the rest of the event data (event tag, ESD and AOD) we do not describe here how it will be distributed between CERN, regional centres and institutes. This is the subject of discussions in Sections 3.7 and 3.8.

The processing power required for reconstructing an event is not yet a well-defined quantity: a reasonable estimate is 250 SPECint95-sec.¹ The total processing power required at CERN for the event reconstruction is estimated to be $7 \times 10^4$ SPECint95. This includes the generation of the calibration and alignment constants, control, monitoring and verification of the data, the first-pass reconstruction and a second reconstruction from the raw data (discussed below), and an efficiency factor of $\sim 75\%$.

The trigger event filter farm will perform general event reconstruction online to reduce the data volume to the offline. There will of course be a certain amount of coordination between the offline and the trigger event filter farm; however, it is too early to discuss this in detail. For a description of the trigger event filter farm see Section 5.2.3.

3.5.1.1 Reconstruction scenario for the initial data-taking

ATLAS will be ready for initial data-taking with a surveyed and aligned detector prepared with the aid of simulation, test beam and cosmic ray studies. The acquisition of real data from proton collisions within the detector will be needed to complete the calibration and alignment and will inevitably expose problems which will require the tuning and development of the algorithms used in reconstruction and analysis. We will establish resources to manage and perform this development, iteratively, on those collections of events, or parts of events which are essential for detector studies.

Those channels with important physics discovery potential will form other collections to receive priority in reconstruction processing. The reconstruction of all data will start as soon as adequate reconstruction results are obtained. This will provide common data for further analysis work to be done from the ESD or AOD level and reduce the I/O and computing resources needed for a given task. A complete reprocessing, with improved calibration, alignment, algorithms and database schemas will be required for the data of this first period. The infrastructure will be sufficiently flexible to allow decisions on the scheduling of the reconstruction of standard samples to be made when the data quality is known.

¹. Note that this represents a factor of $\sim 10$ over the processing power required per event for CDF and D0.
3.5.1.2 Reprocessing: full and partial

By reprocessing we mean the generation of a new version of the ESD for a certain set or subset of events. We define here two different types of reprocessing, depending upon their starting point: full reprocessing starts from the raw data, whereas partial reprocessing starts from the ESD and regenerates at least part of the ESD.

The conditions will evolve rapidly during the first data-taking periods as the detector is better understood. At the beginning it is expected that one full reprocessing of the data collected each year will be needed, which may be performed in parallel with the data-taking. It is not considered reasonable to foresee resources for more than one reprocessing of the ~1 Phyte of data taken each year. After the first few years, the understanding of the detector is expected to have stabilized and it may not be necessary to reprocess all of the raw data.

It is expected that a certain amount of reprocessing can be done starting from the ESD. For example, one should be able to refit tracks from the inner-detector hit information. What is implied here is that there are raw data objects that have been ‘clustered’ with the ESD information\(^1\). Starting the reprocessing from the ESD has the advantage of reducing the amount of data required to be accessed and/or the amount of processing power when one wants to apply small but significant changes to the reconstructed data which arise, for instance, from changes in the calibration, alignment, means of track fitting, etc. It is estimated that the ESD represents ~100 kbyte/event, or 10% of the raw data size. Since a detailed study has yet to be done to justify this in detail, it has been estimated conservatively based on past experience. So we conclude that work remains to be done in this context.

In a distributed computing model scenario with regional centres, the ESD is the proposed basic unit which will be replicated to the different centres. In this case, the partial reprocessing can be shared between CERN and the regional centres: each centre could perform a part of the reprocessing and distribute by network or other transfer media the resulting new version of the ESD.

The request for the partial reprocessing for a particular sample of events should be motivated by the needs of the analyses on that sample. It is also clear that there should be a rapid response to the requests for reprocessing proposed by the individual physics groups. However, there is a need for coordination of the partial reprocessing at the collaboration level because

- any particular event may be flagged to be in more than one logical physics stream, thus more than one physics group may be interested in the same events and hence by the reprocessing;
- other physics groups would often also be interested in updating their ESD when, for example, the calibration or the standard means of track fitting changes.

It will also be necessary to have a general mechanism which will allow physicists to easily understand, for each reprocessed version of any sample, which calibration and alignment constants, reconstruction algorithms, etc. were used.

---

1. In the classical approach the hits used by tracks would be copied from the raw data to where the ESD is stored. From an ODBMS perspective, one would not a priori need to copy the hits: a simple association from each track to its hits would be possible. However, when one comes to optimize the reprocessing starting from the ESD, one may need to recluster and/or copy the hits of each track. There would also be implications if one were to replicate the ESD to a set of regional centres.
3.5.1.3 Generation of the AOD

The Analysis Object Data is the information that a physicist would most frequently need to access during analysis. The AOD will initially be generated during the event reconstruction after Level-3. Then, whenever there is a reprocessing of the ESD, a new version of those AOD objects which depend on the changes will have to be generated.

3.5.2 Monte Carlo production and analysis

Monte Carlo simulation is an indispensable tool in every physics analysis. It is the basis for detector design work, understanding the performance of the detector prototypes and of the complete detector. The present version of the ATLAS full detector simulation, DICE (see Section 4.2.5.3), is based on the GEANT3 simulation package. With the ATLAS software moving to object-oriented tools and languages, the new simulation will have to be written when GEANT4 (see Section 4.6.4.1), the new OO version of GEANT3 becomes available.

A complete but slow Monte Carlo simulation is usually accompanied by a fast version of the detector simulation, in which parametrization of detector response, rather than fully detailed description, is used. Tuning both the slow and the fast Monte Carlo simulations, is a major task in any experiment. Test beam studies, and later dedicated studies using special samples of real data, are especially important here. The data, whether real, simulated or coming from test beams, should have the same structure. By comparing the results of analyses based on these three types of events, users may both tune the details of the Monte Carlo physics and fragmentation models, and verify and tune the detector simulation programs.

Monte Carlo data with the same structure as the real data is used in software algorithm development and for the verification of the reconstruction and analysis programs. Such simulated data can be used to test elements of the real detector, for example the trigger and data acquisition systems, which are downstream of the front-end electronics.

ATLAS must provide sufficient computing capacity to generate and reconstruct the necessary Monte Carlo events during all stages of the experiments, from the design stage, through the construction until the data-taking and analysis. From past experience, one estimates that the number of events which will have to be generated is about 1/10 of the total number of events in the expected real data set. Given that 7.5x10^3 SPECint95-sec is required to generate a single event with DICE, the CPU power of about 5x10^4 SPECint95 will be required to generate such a sample of events in 6 months time. Obviously, such a large load (comparable to that of reconstructing the entire data sample) could be spread among the collaborating institutions, and a large fraction of the Monte Carlo simulation could be done in regional centres. Clearly not all events would have to be generated with the complete (slow) detector simulation programs.

The Monte Carlo databases of the resulting ESD could be replicated to those regional centres who request it, whilst Monte Carlo events in raw data format, with all details, could remain in the regional centre where the set was generated. The availability of the necessary CPU, storage, as well as the co-ordination of the Monte Carlo simulation activities at CERN and other institutes in the world must be a very high priority item for the ATLAS Collaboration.

In Figure 3-3, an overview of the interconnections between analyses of test beam data, real data and their connections with the Monte Carlo simulated data is presented.
3.5.3 Physics analysis

We define an analysis scenario to be the activity where a physicist is working to extract a signal and/or to make measurements for a specific physics channel.

This activity can be considered as an iterative process of perfecting a set of selection criteria composed of two parts:
1. Looping over the sample of events of interest, applying a current set of selection criteria and producing output which can be viewed interactively.

2. Histogramming the results of 1. to understand the effects of the selection criteria on the data, and to improve/vary the criteria.

It is assumed that the organization of this activity will be divided between a central coordinating group and the various physics working groups.

It should be noted that there are closely related activities to what is defined as analysis above which concern the understanding of the detector, e.g. calibration and alignment, as well as the development of the reconstruction algorithms. In general, these activities will often need specialized samples of events and more often access to raw, as well as, reconstructed information. These subjects will not be treated in further detail here.

3.5.3.1 Access to the event information for analysis

We assume that for analysis purposes, the AOD information is sufficient to work with. Hence this will typically be the event object group (see for definition Section 3.2) which is made available to the physicist in order to limit the resources required to access the data. Implicit in this assumption is the fact that the ESD and the AOD information will need to be updated as the calibration, reconstruction algorithms, standard selection criteria etc. change.

If the AOD information is not sufficient for analysis under all conditions, access to the ESD information may be required. During the early phases of the experiment, where the understanding of the detector changes rapidly, it may be preferred to allow access to a frequently updated 'certified set' of calibration constants and reconstruction algorithms rather than regenerating the AOD information frequently. This question of accessing the data during turn-on will have to be studied in more detail. It is clear that for small samples of events, access to the ESD information does not make large demands on the system resources. However, for large samples, the increase in volume by a factor of 10 of the ESD information over the AOD can significantly alter the system load. For the moment, we assume that the AOD information will be the basis for estimations of data access requirements.

After selecting events according to their AOD information, some analyses require access to more and more detailed information to fully understand the events. This should always possible by navigating from the AOD information to the ESD or raw data for each event, as illustrated by the example of Figure 3-1.
3.5.3.2 The analysis cycle

The analysis cycle described in Section 3.5.3 can be viewed as the dataflow shown in Figure 3-4.

![Dataflow Diagram]

**Figure 3-4** Dataflow for analysis.

Here a user develops sets of private selection criteria and applies them to the AOD physics subset of interest. Typically this is done in two steps:

- first a private AOD sample is generated\(^1\) from a standard sample using some loose criteria,
- these criteria are then further developed through an iterative cycle which improves the selection criteria through their application to the private AOD data and the subsequent examination of resulting distributions.

During this iteration the private AOD sample can be further reduced, as indicated by the two-way data flow between the AOD data store and the 'select and generate histos' process. Periodically the user must regenerate his/her private AOD sample because there are more events and/or the reconstructed AOD quantities have changed.

---

1. There are two questions here: Firstly, what is a private AOD sample? For each selected event this may be the same objects as the standard AOD and/or additional derived information considered necessary to be saved by a user. Secondly, what does it mean to generate a private sample? If one only considers standard AOD objects, then each user can save references to these objects, have them in a local database cache, or copy them into a private database. This is a question of access optimization. Anything new that is generated would be created in a private database.
3.5.3.3 Sharing of the analysis result

Clearly the results of an analysis will be shared and discussed in the standard way through presentations in physics working groups. However, it may be useful to be able to exchange or make publicly available the AOD samples which have resulted from a user’s selection. This could be done by sharing ‘access’ information to the events of interest: e.g. a list of event tags, or a collection of event ‘pointers’. This exchange will also need the status of the event objects to be included in order to clearly identify which version of reconstruction algorithm, and calibration, alignment, and other steering parameter data was used. By this mechanism, any event object can be exactly reproduced.

3.6 Tools for remote communication and collaboration

ATLAS will require the efforts of 1600 physicists and engineers from approximately 150 institutes from 31 countries. Coordination and effective collaboration between these dispersed co-workers will require full use of all the available modes of communication. In addition to the now standard use of computer messaging, Web pages, telephone (including FAX) and physical travel by air or other means, the collaboration must learn to use emerging tools such as videoconferencing and other collaborative environments which are now in infancy or still on the drawing boards. A summary and evaluation of these collaborative tools is included in this Computing Technical Proposal because we believe that new forms of tools will be based on computing and networking and that their emergence and use must be taken into account when designing our computing and networking environment.

3.6.1 Network infrastructure

Computer networks provide much of the ‘glue’ for an extended collaboration such as ATLAS. Virtually no aspect of the collaboration’s work proceeds without exchanges of electronic mail or computer files, often with the help of the World Wide Web. The communications infrastructure which makes these essential tools possible comes from a combination of network links provided for high energy physics specifically, for research and education purposes, and for general use by all internet users. During the past five years or so, the TCP/IP suite of protocols originally developed for DARPA in the US has become the almost universal means of computer networking in the research and education area, and in HEP in particular. Newer protocols are now emerging to enable efficient use of faster physical links, including ATM, Frame Relay, and others. Direct use of these new protocols will bring some advantages, but the most important parameter of the networks is the available bandwidth provided. The vast majority of computer networks used by HEP researchers are connected into the network of networks, generally known as the Internet. The result is all but universal connectivity between the laboratories and universities involved in ATLAS.

What is not universal is the available bandwidth or other measures of performance between any two collaborating institutions. Performance is governed by installed bandwidth capacity, but also by competition with other users of the network and by Acceptable Use Policies of the owners of the networks which may force non-optimal routing on a particular communication. Until a few years ago, standard connections have been T1 (1.5 Mbit/s) or E1 (2 Mbit/s) speeds for long distance ‘backbones’ and slower connections to universities or group local area networks. At the present time, T3 (45 Mbit/s) backbones are standard within many countries, with

---

1. See Section 3.4.1 on page 15 for a discussion on the evolution of network technology.
T1 or E1 local connections. However, links between countries are usually not given the same priority or funded from the same national sources, and are only now beginning to depart from the T1/E1 norm.

The exponential rise of internet use by the general public makes general use connections not reliable enough for routine use by ATLAS. Specific connections for R&E or even HEP provide better service, but there will need to be a careful study of the possible need for specific network connections, especially internationally, to support the high use that will be required by ATLAS and the LHC programme in general.

An initial idea of the necessary bandwidth can be found from the ‘ballpark’ numbers of Table 3-4, which estimates the bandwidth needed for a variety of uses.

<table>
<thead>
<tr>
<th>Network function</th>
<th>Year</th>
<th>Bandwidth/user (Mbps)</th>
<th>number of users</th>
<th>Total bandwidth (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>1996</td>
<td>0.01</td>
<td>500</td>
<td>5</td>
</tr>
<tr>
<td>File transfer</td>
<td>1996</td>
<td>0.03</td>
<td>500</td>
<td>15</td>
</tr>
<tr>
<td>Videoconferencing</td>
<td>1997</td>
<td>0.2</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>Advanced tools</td>
<td>2000</td>
<td>1.0</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Analysis</td>
<td>2005</td>
<td>2.0</td>
<td>500</td>
<td>1000</td>
</tr>
</tbody>
</table>

3.6.2 Videoconferencing

Videoconferencing is a ‘new’ collaborative technology. It has been used in the HEP community since about 1990, but its acceptance as a universal collaborative tool has been limited by a combination of cost and rapid technical development which in turn has prevented the establishment of a standard that all potential users could adopt. Though the technology is still subject to rapid change, there are presently two major forms of videoconferencing which, at least in general use, do not interoperate.

3.6.2.1 Codec videoconferencing

Better developed and in longer use is videoconferencing with dedicated equipment known as a codec (compression/decompression). These systems are sold by roughly four companies worldwide. They are generally installed as a dedicated set-up in a conference room accommodating 10 or more persons at each location. Conferences are either point-to-point between two such installations or involve multiple conference room set-ups with the aid of an additional piece of equipment called a Multi-point Control Unit (MCU). Although an interface to the internet is possible, these systems are presently connected to each other by the commercial digital telephone service called ISDN. Typical costs are $25,000 to $60,000 to outfit a conference room, and $30 (domestic) to $300 (international) per hour for each connection. This form of videoconferencing is relatively mature and adheres to international standards so that equipment from different manufacturers can interoperate. The high cost of connections has limited the HEP use of these systems to typical bandwidths of 128 kbit/s, providing an audio channel and 5-10 frames per second of full-screen video. In this document we refer to this type of videoconferencing as ‘codec-based’.
Particular attention needs to be paid to multi-point conferences, which make up the great majority of those used in high energy physics because of the large collaborations that cooperate on detector subsystems and sub-subsystems. The required MCU device costs $50,000 or more depending on the number of ports and other features. Equally important is a convenient and efficient scheduling system that takes account of both MCU ports and conference rooms as potentially scarce resources. Most physics videoconferences so far which involve one or more locations in the US make use of the ESNet videoconferencing Service located at Lawrence Berkeley National Laboratory (moved recently from Lawrence Livermore National Laboratory). This provides both the MCU and a Web-based scheduling service that accepts requests from users and determines availability of needed ports and rooms. Telephone companies are beginning to offer the MCU service for a fee, but without the important scheduling service. Some telephone companies require booking as much as two weeks in advance, but this can be expected to change as the service becomes more common.

3.6.2.2 Workstation videoconferencing

A newer form of videoconferencing adds a video camera and sometimes other dedicated video hardware to a general purpose computer workstation (initially UNIX computers, but now also IBM or Apple types of personal computers). Within the HEP community, the versions of these tools used have been developed by computer science researchers, especially at Lawrence Berkeley National Laboratory and at Xerox Research Labs, and made available without charge. Tens of such systems have also been developed by commercial software houses. Research development of videoconferencing software continues, however, because of the commercial focus on one-to-one conferencing and lack of attention to issues of graceful scaling with distance and with the number of participants. At present, these multiple videoconferencing systems are proprietary and do not interoperate. In this document we will refer to this type of videoconferencing as ‘workstation-based’.

Costs of workstation videoconferencing are generally more modest than for codec systems: the hardware costs $1000 to $5000 on top of the $5,000 to $20,000 cost of a computer workstation. Performance is more modest, with 128 kbit/s supporting an audio channel, and only 1 to 5 frames per second of 1/4 of a full TV screen. Connections are via the internet, using the native network connection of the workstation. This makes the connection ‘free’ except that installed internet bandwidth often is not adequate to support one or multiple videoconferences and performance can be poor. At the moment, workstation videoconferencing is generally more suitable for individual use than for use in a conference room with a group of participants. General adoption of these tools has been slowed by the lack of the necessary bandwidth internationally and also by the developmental nature of the products, requiring considerable expertise to install and manage. A particularly troublesome requirement with the present IP networks is the need for reliable packet delivery without retries. Both audio and video streams are only usable if packets arrive in the correct order and without excessive delay. In the standard internet world, videoconferencing packets will compete on an even footing with other traffic and congestion is always a possibility. Newer protocols such as ATM, or proposed modifications to the Internet Protocol, permit bandwidth reservation or other schemes to provide the needed quality of service.

Workstation videoconferencing can be expected to be given much attention in development from both computer science communities and commercial vendors. Codec conferencing, on the other hand, is relatively mature and will develop more slowly. To the extent that codec videoconferencing continues to be used, ATLAS, or the CERN community in general, will need to consider ways to provide effective MCU access and scheduling services. As both workstations and algorithms become more powerful, we expect that workstation conferencing will overtake
codec conferencing and become the standard form of videoconferencing. We can also confidently expect that standards (perhaps those presently employed in the codec world) will emerge to allow the multiple systems to work together. It seems clear that ATLAS must plan for the general use of videoconferencing and other related tools (see below) on its computer workstations. Some members of the collaboration are already using these tools and we expect (with considerable uncertainty in either direction) that their use will be pervasive within about five years. The ATLAS computing group believes that these tools will be as crucial to the ATLAS-sized collaboration as email and file transfer tools were to the Tevatron/LEP generation of particle physics experiments. We must follow the development of the technology and become early users as they mature.

As a footnote, we take note of the developments within the community to make the two types of videoconferencing interoperate. Especially active are the Caltech group and the HEP Network Resource Center group based at Fermilab. These are important interim tools that will be important to ATLAS while both styles of videoconferencing are in use.

### 3.6.3 Advanced collaborative tools

Computer-based tools for collaborative work at a distance are being developed rapidly both in the research community and in industry. Workstation-based videoconferencing is an early version of what can be expected to develop into a rich environment for collaborative work. In addition to videoconferencing, there is already available (usually bundled together) the facility of a shared whiteboard, a common window on the screens of multiple users on which graphics or text can be displayed and annotated by each user separately. Further tools are being developed commercially for business users. In addition, the US Department of Energy (DOE) is beginning a major initiative over the next five years to develop the vision of a ‘collaboratory’, defined as “an open laboratory... spanning multiple geographic areas where collaborators interact by electronic means” (see [3-6]). Expected capabilities from DOE2000 and comparable projects include immersion environments with a greater sense of presence than current videoconferencing efforts, remote access to instruments and equipment, not just to pictures and sound, shared virtual reality visualisation of inaccessible environments, including detector components, and simulated and real detector data and comparable developments aimed at creating the illusion of ‘being there’.

We believe that these developments will be important for the large-scale collaboration that must come about for ATLAS to become reality. Current plans are for these capabilities to develop on the same time scale that ATLAS has for construction and commissioning. In order to take advantage of these emerging capabilities, ATLAS will need to follow the development and take aggressive steps to become early testers and users of these systems. It is probably not practical for ATLAS to take a major role in the development of these new capabilities, but we should be ready to use them as they become available and ready to work with third party developers to ensure that our requirements are addressed in their development. It should be noted that these new tools will place an even larger burden on the network infrastructure if they are to be used effectively.

A major conclusion here is that the network is crucial to ATLAS and we must understand what capabilities will be provided by the community and what, if anything, must be provided especially for ATLAS.
3.7 Variations of the system architecture

We address here the question of what the possible variations in the roles of the three main actors in the offline: CERN, the regional centres, and the home institutes. We recall the basic offline tasks:

- calibration and alignment of the detector, and development of reconstruction algorithms
- reconstruction of events coming out of the Level-3 trigger
- reprocessing of events: regenerating ESD from the raw data and ESD from themselves
- Monte Carlo generation
- physics analysis

One can consider a wide spectrum of possible models, from keeping all data at CERN to distributing the raw data. We propose here only two models: a centralized model and a partially decentralized model. Because of the vast amount of data that will be produced each year (~1 Pbyte), we propose that the event reconstruction is performed at CERN and that the bulk of the data remains at CERN.

In the centralized model all of the event data will be stored centrally at CERN and are managed coherently. Assuming the use of an ODBMS, to first approximation there is no need to physically copy event information: different subsamples of events can be just *virtual subsets* where the same physical event appears in more than one subsample.\(^1\) The user interface, in particular for off-site interaction with the system will need to use a client/server technology which provides a responsive interface.\(^2\) For analysis, jobs may be developed locally on one’s own workstation, but will execute at CERN. The amount of data which can flow between the user and the central server will be constrained by the available WAN bandwidth. One can imagine that a minimal scenario would be job submission with returning only the histograms. Beyond this it might be possible to transmit small data samples which, for example, result from analysis queries.

The location of the generation of Monte Carlo events in the centralized scenario can be more flexible. The important parameter is the processing power: \(-5\times10^4\) SPECint95 is needed to generate \(10^8\) events in about 1/2 year. Thus, a distributed generation of events in regional centres or institutes organized from CERN is feasible. The resulting ESD databases will then be replicated back to CERN via networks or tape: \(10^8\) events amounts to ~10 Tbyte of data.

In the partially decentralized model, data analysis would be carried out at CERN and at approximately five regional centres.\(^3\) As with the centralized model, a responsive user interface is needed for interaction with the regional centres. From an ODBMS point of view, data would be replicated from CERN to the regional centres. This would mean that all copies of the data would be in the same federated database, and changes to one copy would be propagated to the other copies by the ODBMS services. The replication could be done by either network, given sufficient bandwidth, or tape. A policy would be needed to decide which data to replicate. For ex-

---

1. It may be that frequently-accessed data will need to be replicated to improve throughput by allowing parallel access to the same data. For more information on replication see Section 3.4.2 and Ref. [3-4].
2. For example, starting an application at CERN on one’s X-terminal at home would be insufficient because each keystroke or mouse action is transmitted to CERN and back incurring an unnecessary latency penalty. Instead, one would prefer a more substantial client interface which only makes use of the network for transferring decent size blocks of information. This could be done, for example, with distributed objects or remote procedure calls.
3. Some of these centres may be physically located at CERN.
ample all ESD data could be replicated, which would be 100 Tbyte of data per year. This model would incorporate a distributed generation of Monte Carlo events as in the centralized model.

Concerning the accessibility of a regional centre in the partially decentralized model, we believe that a policy should be developed to allow open access to the services of the different regional centres of the whole of ATLAS.

The role of the home institute will essentially be the same in both the centralized and decentralized models. The main difference being that some physicists will be using the data access facilities of a regional centre as opposed to the central facility at CERN. Each home institute will provide the desktop infrastructure and possibly a small amount of storage and processing power for transmitted data samples. The home institute must also provide the network connection to the regional centres, if they exist, and to CERN. An estimate for the minimum bandwidth\(^1\) required only for physics analysis is: for each physicist performing analysis \(\sim 1\) Mbps, which also allows for transfer of information for event display. Estimates for networking requirements for collaborative work are given in Section 3.6.

### 3.8 Evaluation

Technology issues are essential ingredients of the computing model. The extrapolations of cost estimates for storage, computing power and networking are not very reliable over a time scale from today to 2005. It seems, however, that the cost of sequential (tape) and random (disk) storage, and the cost of computing power can be reasonably expected to decrease to the point that one could satisfy the requirements of reconstruction and analyses within an affordable budget. The largest uncertainty is in the future availability of WAN bandwidth. The two basic models being considered in this document allow a wide range in network bandwidth. No decision is expected in the near future. The raw data reconstruction will be carried out at CERN, and CERN will also be a repository of the central ESD database. The two variations of the computing model architecture described in Section 3.7 are the:

1. centralized model
2. partially decentralized model

In order to illustrate the unavoidable limitations caused by key technological factors, we will consider the following values of some of the parameters discussed earlier.

- number of analysis groups: 20;
- number of members in an analysis group: 10-30;
- number of 'effective' simultaneous interactive users: 150.

We consider three modes of analysing the data:

1. Each analysis group accesses the entire data set \((10^9\) events\) once a month, creating a common sample of about 1-10% events in the total sample. These analyses do not require much processing power. We assume 0.1 s/event on a 2.5 SPECint95 machine. To allow all groups to obtain their results in a week, one needs:

\[
(0.1 \text{ s/event}) \times (2.5 \text{ SPECint95}) \times (10^9 \text{ ev}) \times (20) / (3600 \text{ s/h}) / (24 \text{ h/day}) / (7 \text{ days}) = 8 \times 10^3 \text{ SPECint95}
\]

\(^1\) The estimate is for a user to have a delivered bit rate of 100 kbit/s. For interactive traffic, it is common to apply a factor of 10 to derive the allocated bit rate.
2. Within each analysis group, physicists analyse the preselected samples. We assume 10 s/event on a 2.5 SPECint95 machine for this type of analysis. The result will be a further reduced data set, or a future equivalent of an Ntuple. With 400 physicists actively analysing data, the estimate of processing power required to perform 15 such analyses within a day is:

\[
(10 \text{ s/event}) \times (2.5 \text{ SPECint95}) \times (0.01 \text{ to } 0.1) \times (10^9 \text{ ev}) \times (15) / (3600 \text{ s/h}) / 24 \text{ h} = 4 \times (10^4 \text{ to } 10^5) \text{ SPECint95}.
\]

3. Physicists analysing their subsets, as defined in 2. To allow 150 physicists to work simultaneously and to obtain their results within 1/2 day, assuming 1 s/event for this type of analysis, the required processing power is:

\[
(1 \text{ s/event}) \times (2.5 \text{ SPECint95}) \times (0.01 \text{ to } 0.1) \times (10^9 \text{ ev}) \times (150) / (3600 \text{ s/h}) / 12 \text{ h} = 9 \times (10^4 \text{ to } 10^5) \text{ SPECint95}.
\]

The total amount of processing power required for analysis is estimated to be at least 1.3\times10^5 SPECint95.

The key element required to make a centralized scenario plausible is the availability of sufficient network bandwidth to sustain simultaneous work by ~150 physicists each requiring reliable and fast response for his/her interactive work. As mentioned in Section 3.7, a minimum bandwidth of 1 Mbps per user is needed for analysis. To allow 150 users to perform their analyses simultaneously, an allocated bandwidth of 150 Mbit/s from CERN will be required. Note that this estimate does not account for other background interactive work, nor does it assume any increase in the required bandwidth to allow for future developments in interactive analysis tools. A factor of 10 increase could easily be required, bringing the bandwidth requirement to 1.5 Gbit/s. Given that ATLAS, CMS and other LHC experiments will run concurrently, such a bandwidth may be difficult to afford. To compare, in the partially decentralized scenario, the WAN bandwidth required to replicate in real time (i.e. at 100 events/s rate) a full year's data set in ESD form, twice a year, to a single regional centre is 0.48 Gbit/s². For five regional centres, assuming that the data will transferred in parallel (which is the most conservative estimate), the bandwidth of 2.4 Gbit/s would have to be allocated. However, if such WAN bandwidth is not available, the ESD databases could be replicated by other means - tape and plane - using WAN to distribute only the AOD or subsets of ESD data.

Another important factor is the data transfer rate from disk to computer memory. To achieve the required analysis time (days for the full ESD, hours for sub-samples of 10⁶ to 10⁷ events in AOD format) the data containers (disks) will have to be replicated in order to provide for sufficient parallelism in data I/O, depending on the actual transfer rate in 2005. In the centralized scenario these replicas would reside at CERN; in the partially decentralized scenario they would be distributed to regional centres. The advantages of the latter solution include reduction (by a factor 1/N) of network bottlenecks at CERN, higher reliability, redundancy, flexibility and robustness of the distributed system. The important rôle of regional centres in sharing the large anticipated load of Monte Carlo simulation has already been recognized. Also, given that 80% of the ATLAS software will have to be written outside CERN, the regional centres not located at CERN would play an important rôle in code development, providing training and programming experts.

1. For example file transfers, computer-supported cooperative work, and videoconferencing/multicast services.

2. An allocated bandwidth of 0.48 Gbps corresponds to a sustained bulk transfer rate of 0.16, i.e. one must allow a 30% efficiency, ~3 times better than for interactive traffic.
Given the expected world-wide distribution of regional centres, it would be highly desirable for all ATLAS users to be able to access all of them, to take advantage of time zone variation.

To first order the computing power and storage costs would be the same in both scenarios. It is only the cost of personnel to run the regional centre and possibly new infrastructure which would constitute the added cost in the partially decentralized model.

Careful modelling of the performance and costs of the two scenarios will be required before a decision can be taken about which scenario to choose for ATLAS computing. It is imperative, however, to realize that no rational decision can be made until the key technical factors are known better than at the present time.

### 3.9 Cost estimates

It should be stressed that the costs given below are preliminary. Both the resources required and the expected unit costs at the time of investment are rather uncertain, such that any cost estimate can vary by at least a factor of two.

It should also be noted that we discuss here the cost of the ATLAS computing infrastructure for the hypothesis of a single central installation at CERN. Any major regional centre would share the costs for simulation and have to add expenses for analysis processing power\(^1\), storage and personnel. Furthermore, the cost estimates given do not include the cost for desktop equipment and the corresponding software for the collaboration members. The cost of desktop equipment is estimated to amount to 100-150 CHF per month and per station.

Personnel to operate the data processing and the data storage systems with a reliability of >99% are required (less than four days of unavailability per year).

An estimation of the cost for ATLAS computing is as follows:

The requirements for processing power is the sum of

- Reconstruction: \(7 \times 10^4\) SPECint95\(^2\);
- Analysis: \(13 \times 10^4\) SPECint95;
- Simulation: \(5 \times 10^4\) SPECint95,

yielding a total of about \(2.5 \times 10^5\) SPECint95.

The disk requirement of 100 Tbyte is based on the assumption that about 10% of the data volume - relative to the raw data volume - needs to be available on direct access media.

The tape requirement of 1 Pbyte/year is given by the expected raw data volume.

The cost estimates given in Table 3-5 are derived by multiplying these requirements by the expected unit costs of 50 CHF/SPECint95 processing power, 12 CHF/GB disk, and 1 CHF/GB automated storage medium (e.g. tape). These expected unit costs are derived from the extrapolations of cost-performance trends to the year 2005, which were given in Section 3.4.1, and have been scaled by a factor of 1.5 to account for the spending profile given below.

---

1. Additional analysis processing power may be required at regional centres to the extent that this requirement is taken from the low end of the estimate given in Section 3.8.
2. For a definition of SPECint95 see page 10.
It should be noted that the software cost estimates pertinent to the computing model have not been included.\[1\] The more general question of software licenses is discussed in Section 4.9.

The extrapolation of performance and cost of wide-area networking seems to be much less evident. Evolution has been comparatively slow over the past years. However, substantial improvements are expected due to deregulation of the European PTTs and due to new demands for networking by the general public. As a price performance prediction is impossible, we have assumed that the spending for external networking at CERN will be roughly at the current level of 1 MCHF, and that outside centres will spend in total about 3 MCHF for their connectivity.

<table>
<thead>
<tr>
<th>Item</th>
<th>Size</th>
<th>Initial Cost[a]</th>
<th>Annual Cost[b]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing power</td>
<td>2.5x10^5 SPECint95</td>
<td>19 MCHF[c]</td>
<td>3 MCHF</td>
</tr>
<tr>
<td>Disk</td>
<td>0.1 Pbyte</td>
<td>2 MCHF</td>
<td>1 MCHF</td>
</tr>
<tr>
<td>Tape</td>
<td>1 Pbyte</td>
<td>1 MCHF</td>
<td>1 MCHF</td>
</tr>
<tr>
<td>Network</td>
<td>???</td>
<td></td>
<td>4 MCHF [d]</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>22 MCHF</strong></td>
<td><strong>9 MCHF</strong></td>
</tr>
</tbody>
</table>

\[a\] This is in addition to a yearly operational cost of ~500 kCHF for the central ATLAS infrastructure at CERN.

\[b\] This is for 2006 and beyond.

\[c\] Note that half of this cost is for analysis and is derived from the lower estimate given in Section 3.8.

\[d\] During set-up this cost will be smaller.

After the initial investments, annual costs arise due to the increasing data volume, necessitating 1 Pbyte of permanent archive storage (tapes) every year, and a corresponding increase in disk space. Processors have a limited lifetime and we assume that they will be replaced by yearly upgrades of 20% of the initial investment.

Although it is desirable to purchase computing equipment as late as possible due to the improving price-performance ratio, the infrastructure needs to be built up gradually providing for a more favourable spending profile and allowing the behaviour of the system to be studied in a scaled down version.

In particular, we propose the spending profile given in Table 3-6.

---

\[1\] In particular there are the ODBMS costs which are discussed in Section 4.9 and the mass storage software costs, where, for example, the current US costs for HPSS, which was discussed in Section 3.4.2, are $300k for the first year, and $150k per year thereafter.
Table 3-6  Spending profile for the central infrastructure.

<table>
<thead>
<tr>
<th>Year</th>
<th>kCHF</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1997</td>
<td>500</td>
<td>Detector optimization, program development, test beam analysis</td>
</tr>
<tr>
<td>1998</td>
<td>500</td>
<td>Detector optimization, program development, test beam analysis, 1% prototype of OO-database</td>
</tr>
<tr>
<td>1999</td>
<td>500</td>
<td>Program development, test beam analysis, database and farm studies</td>
</tr>
<tr>
<td>2000</td>
<td>800</td>
<td>Program development, test beam analysis, database and farm studies</td>
</tr>
<tr>
<td>2001</td>
<td>1000</td>
<td>Program development, test beam analysis, develop processing farm</td>
</tr>
<tr>
<td>2002</td>
<td>2000</td>
<td>Program development, test beam analysis, 1% prototype of processing farm</td>
</tr>
<tr>
<td>2003</td>
<td>4000</td>
<td>Program development, test beam analysis, 10% prototype of OO-database</td>
</tr>
<tr>
<td>2004</td>
<td>8000</td>
<td>Software completion, final tests, implement full size database</td>
</tr>
<tr>
<td>2005</td>
<td>10000</td>
<td>Full processing farm, full ‘tape’ storage, running-in of system, data-taking</td>
</tr>
<tr>
<td>2006 and each subsequent year</td>
<td>10000</td>
<td>Data-taking, software maintenance, adaptation to new requirements, hardware exploitation and maintenance</td>
</tr>
</tbody>
</table>

3.10 Milestones

The key steps in the construction of the ATLAS offline system are

- proof of principle that an ODBMS can satisfy the requirements for the event reconstruction and analysis systems;
- demonstration that a commercial mass storage system which can satisfy our requirements can be interfaced as the transparent back-end of an ODBMS;
- movement from the current FORTRAN/ZE BRA based offline system to one based on object-oriented concepts, C++, and an ODBMS;
- deciding on the number and rôle of regional centres;
- progressive construction, in terms of both hardware and software, of the event reconstruction and analysis systems in coordination with the regional centres.

Table 3-7 presents a set of proposed milestones to carry out the above steps. The proof of feasibility of the ODBMS is currently being carried out by the RD45 project, which expects to have carried out extensive tests on a few Tbyte system (disk + tape mass storage) by the end of 1997. At this time we will be able to decide whether or not to retain a commercial ODBMS with the capabilities described in [3-4] as the ATLAS baseline option for data management. After 1997 it is expected that this concept will begin to be used for experiments, for example Babar (few tens of Tbyte) in 1999 or COMPASS\(^1\) (100 Tbyte) in 2001. For ATLAS, it will be important to move to the use of an ODBMS system by the end of 1998. The construction of the final system will begin

---

in 2001 with the bulk of the purchases delayed as long as possible to benefit from the continuous drop in hardware costs.

### Table 3-7 Milestones for the development of the offline computing system.

<table>
<thead>
<tr>
<th>Date</th>
<th>Milestone Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>End 1997</td>
<td>Proof of the feasibility of the federated OO-database concept</td>
</tr>
<tr>
<td></td>
<td>(the mass storage interface to an ODBMS will be demonstrated in 1998)</td>
</tr>
<tr>
<td>Mid 1998</td>
<td>Test of prototype of regional centre model with interaction between CERN and a site</td>
</tr>
<tr>
<td></td>
<td>outside Europe</td>
</tr>
<tr>
<td>End 1998</td>
<td>Provide ~1 Tbyte working prototype of database</td>
</tr>
<tr>
<td>End 1998</td>
<td>Decision on rôle, size, number and location of regional centres</td>
</tr>
<tr>
<td>End 2002</td>
<td>Provide 1% prototype of event-processing farm</td>
</tr>
<tr>
<td>End 2003</td>
<td>Provide 10% (100 Tbyte: disk + robots) working prototype of database</td>
</tr>
<tr>
<td>End 2003</td>
<td>Provide 5% prototype of event processing and analysis farms</td>
</tr>
<tr>
<td>End 2004</td>
<td>Provide 100% database (disk + robots, not necessarily all tapes)</td>
</tr>
<tr>
<td>End 2004</td>
<td>Provide ~40% of final event processing and analysis farms</td>
</tr>
<tr>
<td>Spring 2005</td>
<td>Provide full event processing and analysis farms</td>
</tr>
</tbody>
</table>

The steps leading to a decision on regional centres will be the following:

- This document should provide the basic input required for modelling of the two scenarios: centralized and partially decentralized models.
- A group has been set up to perform detailed modelling, which may be done in collaboration with CMS. It is expected that iterations will be needed to refine the input.
- This same group will also begin working more directly on database questions which concern ATLAS. As a consequence, the interaction and information exchange with RD45 will be strengthened.
- The various physics communities will be consulted so that non-technical issues are also taken into consideration.

### 3.11 References


3-2 PASTA - The LHC Technology Tracking Team for Processors, Memory, Architectures, Storage and Tapes, Status Report, August 1996.

3-3 The LHC Networking Technology Tracking Team, Status Report, October 1996.


3-5 See: RD45 proposal: CERN/DRDC/94-30, DRDC/P59; their web home pages:

http://wwwwnl.cern.ch/asd/cernlib/rd45/index.html; and references [3-1] and [3-4].

3-6 Kouzes, Myers, and Wolf, IEEE Computer Vol. 29, No. 8 (August, 1996).
4 Software

Here we describe where the ATLAS software is now, and where we think we want to be going for 2005. We take this to be Object-Oriented (OO) software and we assume that we will start developing this in C++. However, 2005 is a long way away and as our general strategy will be to follow industrial trends so as to be able to benefit fully from the products in the computing industry, we cannot really say with any confidence what our software will look like in 2005, we can only identify the beginning of the road.

Where we are starting from
We have a detector simulation program based on the GEANT 3.21 detector simulation package and a reconstruction program for the simulated data. All this code is written in FORTRAN 77 and uses ZEBRA for memory management. These programs have been used in the past to study the detector behaviour and to optimize its parameters and have produced all results for the ATLAS Technical Proposal [4-1] and will be used for the preparations of the Technical Design Reports for the various subdetectors. It is foreseen to use and upgrade these programs for at least the next two to three years.

Where we think we want to go
A major fraction of this chapter will describe new software which we propose to develop which will start from the same algorithms, improving them where necessary. We will follow an OO design, implementing in C++. The data, now described in common blocks and stored in ZEBRA banks, will then be encapsulated in objects together with the functions acting on that data.

Structure of this software chapter
Section 4.1 identifies some general requirements for ATLAS software. The next section describes algorithms and techniques which will most likely be maintained in the future from our current software. Section 4.3 lists in general terms OO and its benefits for HEP. Section 4.4 describes how we propose to produce the software, a software process. The next section refers to a document describing the software development environment which we think we need together with a list of choices which have been made so far. The last sections of this chapter describe our strategy for change, our training plan to ensure that we are able to do what we set out to do, some milestones, and a very rough estimate of what the software will cost.

4.1 Requirements for ATLAS software

Software must be provided for the various components of the ATLAS system. Conceptually, we may distinguish several distinct software subsystems: the online software and the offline software, including reconstruction, simulation and the analysis framework. These must all work together to provide the facilities that physicists need.

4.1.1 Major requirements

We list below the major requirements that the software as a whole must have. We divide them into essential physics requirements, which say what the system must be capable of doing, user control, which says what facilities will be provided for the user to control the software, and environmental requirements which describe the practical constraints under which ATLAS software will be developed and run.
4.1.1.1 Functional requirements

Efficient filtering of the online data to select events of a type specified by the user. There are hard real-time constraints on pattern recognition in the Level-2 and Level-3 trigger subsystems as well as in the full (offline) reconstruction of what passes the Level-3.

Simulation of the behaviour of the detector subsystems under different kinds of physics events.

Reconstruction of tracks from detector digitisations.

Particle identification.

Support for the physics analysis of processed event data.

4.1.1.2 User control

Control of the operation of the online system. This will include the selection of the subdetectors to be studied, the setting of operational parameters, and selection of Level-2 and 3 algorithms and their internal parameters.

Control of the processes of simulation, reconstruction, and analysis. This will include the selection of the subdetectors to be studied, the selection of algorithms and their parameters, and the selection of event types.

An easy-to-use graphical interface to allow the user to exercise the options above.

Facilities to display graphically the results (including intermediate results) of subdetector operation, pattern recognition, and reconstruction.

4.1.1.3 Environmental requirements

The software must be capable of being developed and run on a large range of platforms and architectures.

Remote working must be possible.

All developers must have access to a common database of event data (which may however be stored in a distributed manner).

4.1.2 How the requirements will be realized—need for OO and the ASP

In practice, there are many similarities between the various subsystems, since they are all concerned with the same entities: detector components, digitisations and tracks, for example. In all cases, it will be advantageous if the similarities between subsystems are reflected in the software, so that a unified conceptual model is developed. For example, if a user wishes to carry out a reconstruction, the functionality should be largely similar irrespective of whether events are generated by simulation or from a real experimental run. The software for the Level-3 trigger will be very similar to that for event reconstruction. Both simulation and reconstruction must model the detector system, although not necessarily at the same level of detail. Obviously, the online software must include facilities for controlling the hardware of the experiment, and the Level-2 trigger system, but apart from this, the user interface should have similar features to those for controlling simulation and reconstruction.
Given these strong similarities and connections between various parts of the software, it is essential that uniform methods of software development be followed for all ATLAS software products. This will aid communication between groups of developers concerned with different aspects of the system and will assist in the interfacing of the various software components.

The ATLAS experiment has characteristics that have implications for the quality of software that is required. Data are discarded by the Level-3 trigger and can never subsequently be recovered. It is essential that the filtering program works correctly, so that interesting events are not thrown away. The time over which the experiment will run is expected to be about 20 years. This means that the people who maintain and extend the software will not be the same as those who originally wrote it. Short-term software developers such as PhD students and RAs will be entering and leaving the project throughout its lifetime. They must be able to get to work quickly, and make changes in the part of the software that concerns them without causing knock-on effects for other people. There are also a very large number of collaborating institutes, some consisting of large groups and some of just one or two individuals. Communication will have to be primarily electronic. It must be possible for individuals to contribute to software development without having much face-to-face contact with other developers.

Taken together, these facts imply a number of requirements for the design of the software.

1. The overall design should be easy to understand, and the individual modules self-contained and well-documented. Specialists, for example subdetector developers, should be able to work on their own part of the system, without interfering with the rest of the system. This means that there must be a clear division of responsibilities and well-defined interfaces between semi-independent groups of developers.

2. The system must be designed for easy maintenance, taking this word in a wide sense to mean both correcting faults and making improvements. As far as possible, faults must be prevented or eliminated before the software goes into production, but those that remain must be easy to discover, so as to avoid wasting time. This means that software modules must be well encapsulated and loosely coupled to other modules, so that a change in one module does not involve changes in other modules. Good encapsulation is also required for tuning up the software in a way that does not affect existing programs.

3. The system must do what the users require of it. Programs must be correct and efficient, both in identifying events of a given type and in doing it within the given time constraints. The system must be easy to use and reliable.

Although these requirements are very general and apply to any ATLAS software, we think that the best way of achieving these objectives is by object-oriented development. A brief description of the main features of the OO approach is given in Section 4.3.2. Here we may just say that it allows good encapsulation and seems suitable for distributed development. Preliminary studies indicate that an OO database will provide a suitable repository that can be accessed in a distributed manner. The work of RD45 is discussed in Section 4.6.4.2.

The requirements for good-quality software are very stringent. OO development should make it easier to check the individual modules and ensure that they do what they are supposed to do before they are put into production. However, a development paradigm by itself, although it may assist in the production of good-quality, easily maintained software, cannot guarantee it. It is therefore necessary to have project management procedures to ensure that all production software meets the required standards. For this reason, we propose a software process—a scheme for the management of the development process that defines standards and procedures. It is not intended that the software process should limit how people develop their software privately. It primarily concerns the way software from any collaborator is integrated in the official production software. This is described in Section 4.4.
It must be recognized that our proposals may involve a substantial change in the way in which software developers work. We think that the benefits will outweigh the disadvantage of the extra effort that some physicists will have to make in adapting to a new development method and programming language. However, we must acknowledge that this effort may be considerable and provide a path for accomplishing the changeover. These questions are addressed in Section 4.6.

4.2 Existing software

In this subsection we describe the essential parts of the existing software. It is expected that these algorithms will be maintained or improved in our software in the future no matter which software technology is used. The current implementation of this software is done in FORTRAN and uses the ZEBRA memory management package. First a global description is given of the algorithms used in the reconstruction of simulated calorimeter, inner detector and muon detector data as well as of the software which is used to simulate the Level-1 and Level-2 trigger performance. The second part of this section describes the programs and techniques which were used for the description of the geometry of the ATLAS detector and for the simulation of data using the GEANT 3.2 detector simulation package as well as the interface package to existing event generators and the one between the existing FORTRAN datastructures and C++.

4.2.1 Calorimeter reconstruction programs

The calorimeter reconstruction in the present ATLAS software is organized in a similar way in the electromagnetic and hadronic sections. An overall description of both the electromagnetic and the hadronic calorimeters can be found in the ATLAS Technical Proposal [4-1] in which also most of the technical terms in the following text are explained. More up-to-date information about calorimetry can be obtained from the web1.

4.2.1.1 Electromagnetic calorimeters

The reconstruction in the electromagnetic (EM) calorimeter proceeds in three steps:

- Matrices are filled with the energies stored by the simulation code, after a cell to cell calibration. There is one matrix per region of uniform granularity. Electronic and pile-up noise can be added at this stage, if wished.

- A temporary matrix covering the full EM calorimeter range is filled with the above energies after mapping of the granularity of each region onto square cells. The energies are summed over the samplings in depth. A search for clusters with transverse energy above a given threshold is then performed by moving a fixed size window in this matrix and by looking for local maxima. The size of the window and the threshold are optimized to obtain the best efficiency of the cluster search and to limit the rate of fake clusters due to noise. They may depend on the luminosity.

- The position and the energy of each cluster are reconstructed by using all physical cells with centre in a window around the directions provided by the clustering algorithm. The energies are corrected for the finite containment of the window and for the response modulations observed as a function of $\eta$ and $\phi$. The position in $\phi$, measured only in the second

sampling of the calorimeter, is corrected for an offset due to the accordion shape of the de-
tector. Positions in \eta are measured both in the first and the second sampleings, are correct-
ed for S-shapes, and are used to compute the particle direction and to estimate the z
position of the vertex.

Algorithms for particle identification are being implemented. They fully benefit from the opti-
mized granularities of each sampling of the EM calorimeter, and use the information from the
tracker and from the hadronic calorimeters.

The EM reconstruction code is being used for the study of the calorimeter performance de-
scribed in the Technical Design Report. It will evolve to include the best algorithms as our
knowledge of the detector improves, will be tested using test-beam data collected with several
calorimeter modules, and will follow the evolution of computing to be ready for the analysis of
LHC data.

4.2.1.2 Hadronic calorimeters

The reconstruction in the hadronic calorimeters proceeds in the following steps:

- As for the EM part, an unpacking procedure stores the energy deposited in cells belong-
ing to a uniform calorimeter region in a given matrix. There is one matrix per region of
  uniform granularity. Energies are stored after a calibration which takes into account the
different sampling fractions of the various calorimeters. The electronic noise as well as the
pile-up noise can be added at this stage, if wished.

- A combined energy matrix with granularity $\Delta \eta \times \Delta \phi = 0.1 \times 2\pi/64$ is then produced. The
  content of each cell in this matrix is filled with the sum of the transverse energies in the
  EM and hadronic sections smeared according to the real cell size.

- The search for jet candidates is based on a sliding window algorithm, which looks for lo-
  cal energy maxima in a $3 \times 3$ cell window, by scanning the whole combined matrix. If a
  window contains a local maximum and its total energy is above a threshold of 2 GeV, then
  such $3 \times 3$ cell cluster is used to define a jet candidate. For each candidate kinematic vari-
  ables ($<\eta>, <\phi>, \sigma_\eta, \sigma_\phi$ etc.) are calculated, using the information from a
  $9 \times 9$ cell window with the same centre as the selected cluster.

- To avoid double counting, each new jet is compared with all jets selected previously. If the
distance between a new jet and one of the previous jets is less than some limit value, then
the jet with the largest energy is retained. In addition, the reconstructed jet is checked to
be symmetric: if the distance between the local maximum and the jet direction is large,
then such a jet is excluded from the analysis.

- For each reconstructed jet the main parameters (energy, transverse energy, pseudo-rapid-
ity and azimuthal angle, angular widths, etc.) are stored. Furthermore the list of all cells
  (both, cells from the calorimeters and cells from the combined matrix) within a cone of a
given size are kept for each jet.

Most of the parameters used for the reconstruction, such as the granularity of the combined en-
ergy matrix, the size of the windows for clusters and jets, the cluster energy threshold, the jet
separation and jet asymmetry thresholds, are to be tuned according to the physics channel be-
ing studied. For example, the default granularity of the combined matrix (which is $0.1 \times 2\pi/64$)
coincides with the granularity of the hadronic calorimeters and is four times bigger than the ba-
sic granularity of the electromagnetic calorimeter. The default value is appropriate for the
standard jet reconstruction procedure. However, the finer granularity of the EM calorimeter can
be useful for more effective jet separation in the EM calorimeter.
4.2.2 The pattern recognition software for the Inner Detectors

The ATLAS inner detector is made up from a Transition Radiation Tracker (TRT) on the outside with layers of silicon strips on the inside. Closest to the interaction region, silicon pixels are used as well. More details and explanation can be found in [4-1] and more recent information is available from the inner detector web pages\(^1\). There are currently four pattern recognition packages being developed for the ATLAS inner detector: XKalman, iPatRec, PixlRec and iPatRec.

XKalman starts the pattern recognition in the TRT using a histogramming method. Results from the TRT are used as pointers for a track search in the precision layers (silicon strips and pixels) using the Kalman filter-smoother formalism. Each track reconstructed in the precision layers is then extrapolated back to the TRT for final reconstruction and fitting.

IPatRec initiates track-finding from space-points in the discrete tracker. It is designed to exploit the superior two-track resolution and relatively low occupancy of the silicon tracking layers. It works in regions of interest defined by ‘seeds’ such as EM calorimeter clusters for electron candidates, tracks found in the muon detector, jets from the hadron calorimetry and Monte Carlo (‘truth’) particles in the case of simulated data. A local helix interpolation between space-points is the preferred method of associating the remaining discrete hits, as it combines the precision of interpolation with the ability to follow catastrophic processes such as electron bremsstrahlung. Extra parameters are included in the track fitting to allow for the effects of bremsstrahlung and multiple Coulomb scattering. A histogramming technique is used to select and resolve the left/right drift ambiguity of the TRT hits added to the track. The primary use of the TRT is to increase the track-fit precision, however, in the case of the discrete tracking efficiency falling below the design specifications the TRT also provides the pattern recognition information needed to reject false space-point combinations. At the moment 30% of the code in iPatRec is in C++ and using the interface described in Section 4.2.5.4. as part of the effort to change to object-oriented techniques and the use of C++ (see also Section 4.6.1).

In PixlRec, emphasis is put on exploiting pixel fine granularity: the pattern recognition is initiated from the innermost pixel layer and goes outwards. The combinatorial search in pixel detectors is completed with outer silicon strip detectors. Extrapolation to the TRT is available thanks to routines developed for the XKalman package. Multiple scattering errors are treated in the final fit with a Kalman filter algorithm also developed by the authors of XKalman. PixlRec is mainly intended to perform precise tracking in small regions (e.g. inside the jet) where the track density is high. The aim is to test an alternative algorithm for pattern recognition, and thus get a more reliable understanding of the Inner Detector performance, especially in b-tagging.

ILPatRec is being developed as an object-oriented pattern recognition for the inner detector. It begins pattern recognition by using a histogramming method in the precision layers, starting at the innermost layer present. This information is used to define initial candidate tracks which are developed in the remaining precision layers using the Kalman filter technique. It is foreseen to include the TRT for final reconstruction and fitting. At the time of writing of this document, iLPatRec is still under development.

4.2.3 The muon reconstruction software

The Muon Drift Tubes (MDT) of the ATLAS Muon Spectrometer are arranged in chambers in three layers around the beam axis: the inner, the middle, and the outer layer. Each MDT chamber contains two superlayers which each consist of 3 or 4 tube layers. An MDT chamber is able

---

\(^1\) http://atlasinfo.cern.ch/Atlas/GROUPS/INNER_DETECTOR/inner_detector.html
to measure a 'local straight track segment' or track vector. The muon trigger chambers are Resistive Plate Chambers (RPC) in the barrel and Thin Gap Chambers (TGC) in the end-caps and are physically connected to the MDT chambers. A package of MDT and RPC/TGC chambers is called a station. The whole of the spectrometer is in a toroidal magnetic field. More information on the Muon spectrometer and the magnetic field can be found on the web1.

The ATLAS Muon system is designed to obtain high-momentum resolution over a large fraction of the $(\eta,\phi)$ space covered by the detector, up to muons with $p_T$ of the order of a few TeV. The Muon Pattern Recognition Algorithm must be designed to match the high quality of the muon spectrometer. In particular the pattern recognition program must cope with three basic points:

1. the high background level present in the ATLAS experimental hall which yields single tube occupancy up to values as high as 30%;
2. the high inhomogeneity of the magnetic field which forbids any assumption on simple analytical shapes for the muon tracks;
3. the variety of the muon chambers used and the complexity of the layout of these chambers, designed to perform the best track reconstruction achievable within the technical and financial constraints.

The strategy of the pattern recognition algorithm, described below, can be summarized in four main points:

1. identifications of 'regions of activity' in the muon system, through the RPC/TGC systems;
2. reconstruction of 'local straight track segments' in each muon station of these regions of activity;
3. combination of tracks segments of different muon stations to form muon track candidates;
4. global track fit of the muon track candidates through the full system.

In a first step, 'regions of activity' (ROA) in the $(\eta,\phi)$ space are identified using information from the trigger chambers. The size of these regions is roughly $d\eta \times d\phi = 0.4 \times 0.4$ and they are centred where there exists at least one RPC/TGC hit in both coordinates (among up to seven layers of RPC/TGC).

Then, all muon chambers intersecting with these ROAs are selected for the muon track reconstruction. Straight track segments are first reconstructed individually, in the bending plane, in each muon station, trials being performed with each MDT hit of a multi-layer in combination with, in turn, all MDT hits of the other multi-layer belonging to the same muon station.

When there are no 'second coordinate' measurements (i.e. RPCs or TGCs) nearby the station, several second-coordinate positions are tried in turn because the transverse position influences the 'first coordinate' measurement through Lorentz angle effects and propagation time of the signal along the wire. The hit pair is required to point loosely to the interaction vertex in order to suppress background tracks and combinatorial background. All four possible tracks candidates of each hit pair are extrapolated to the remaining tubes of the MDT station and matched with the recorded hits to possibly validate the trial (and remove the four-fold ambiguity). Track segments are selected if they have at least two hits associated in each multi-layer and a sufficiently 'good' $\chi^2$ of the associated linear fit in which the following effects are accounted for:

• degraded single-hit resolution in presence of delta rays

Tube hits expected with a TDC time larger than the one recorded are weighted much less than the opposite case, if \((T/TDC - Textr.) < Tdead\), where Tdead is the time interval during which the electronics remains insensitive to the pulses produced by additional successive tracks.

• detection efficiency of each individual MDT tube

The better the tube efficiency is expected to be, the higher is the maximum contribution to the \(\chi^2\) of tubes which are crossed (far enough from the tube wall - where efficiency drops) by the trial segment and which do not have a hit (or a hit far 'behind' the predicted position).

In a first pass, this straight track segment search in the transverse plane is performed only for outer and middle stations in the barrel and inner and middle MDT stations in the end-cap, i.e. where second coordinate chambers are available. In this first search which is called 'strict search', track segments found in the bending plane must be associated with at least one second-coordinate hit.

A looser independent track search follows the above one, based on less stringent \(\chi^2\) requirements on the candidates and without any request on the matching with second-coordinate hits. As in the strict search, all available hits belonging to the ROAs are used in this second search, called 'loose search'.

This loose search is completed by a third class of straight track segments: those crossing only one of the two multi-layers of the station. Here, each multi-layer of the station is used individually to identify triplets (or quadruplets) of tubes with hits compatible with a straight track. In this last case, in order to reduce the large number of possible combinations, only the hits left unused by the track segments found in the previous searches are considered and the \(\chi^2\) requirements are strict.

The position and direction of 'strict' track segments found in the outer and middle stations of the muon spectrometer allow a first rough estimate of the momentum of corresponding candidate muons. Each of these strict track segments is then extrapolated to the first middle or outer (or even inner when middle or outer is missing) station found using tracking in the magnetic field, several trials being performed for different values of the momentum around the first rough estimate ('momentum scan'). If there exists some matching, in position and direction, of the extrapolated tracks with one (or several) 'loose' track segment(s) in this next station, the one with the best matching is included in the candidate track and a fit is performed leading to a second and more accurate estimate of the momentum. In this fit (and the following ones) full tracking is performed at each step of the minimization procedure. Then, a second and finer 'momentum scan' around the improved momentum estimate is performed with extrapolations to all other potentially crossed stations. Any matching loose track segment in these stations is included in the candidate track. After this stage, a candidate track is kept only if it contains at least two tracks segments. Using all the track segments belonging to the candidate track, a new fit is performed to fine tune its position, direction, and momentum.

Finally, a last global fit is performed, starting from the best result of the previous fits, but using this time directly the raw information available information (i.e. the TDC values and hit strips instead of the pre-reconstructed straight track segments). The purpose of this last stage is to get a global and more realistic estimate of the likelihood of the candidate track and, for example, to be able to select, among all the hits a priori belonging to the track, the 'good' ones from the 'bad' ones (those spoiled by delta rays or gamma or neutron background) that lay too far away from
the reconstructed path of the muon. Eventually, the selection of reconstructed muons is made according to the value of the $\chi^2$ of this last global fit.

### 4.2.4 Simulations of algorithms for LVL1 and LVL2 trigger

The ATLAS trigger is organized in three levels (LVL1, LVL2, LVL3). At LVL1 special-purpose processors act on reduced-granularity data from the calorimeter and muon subsystems. The LVL2 trigger uses full-granularity data from most of the subsystems, starting with data from regions of interest (RoI) identified by the LVL1 as containing interesting information. At LVL3 the full information is used to make final selections.

The LVL1 trigger accepts data every 25 ns (40 MHz); the trigger decisions have to be formed and distributed within 2 $\mu$s. The maximum output rate is limited to 100 kHz. The average LVL2 decision time is limited to about 10 ms and the output rate to a few kHz. This puts stringent limits on the processing that can be done by trigger algorithms.

The trigger simulation package, called ATRIG, supports development and evaluation of trigger algorithms. This is a feedback process where detector layout and trigger hardware impose limitations, but in turn the hardware is influenced by the results of the simulation studies. The criteria for evaluating an algorithm are efficiency for the expected physics signal and rejection power against the unwanted background. In addition the trigger has to be open for unexpected physics. The algorithms as they were used for the Technical Proposal [4-1] are summarized in [4-1] and [4-2], which contain references to the detailed algorithm write-ups. Performance of the algorithms has to be studied for low and high luminosity. At high luminosity, there will be about 23 interactions in each beam crossing. The simulations of algorithms, trigger decisions and the resulting performance of the trigger will be presented in full detail in the TDR for the trigger which is due at the end of 1997.

Three types of algorithms are presented in this section: LVL1 algorithms, LVL2 feature-extraction algorithms, and LVL2 global-decision algorithms. LVL3 algorithms are similar to offline algorithms and will not be discussed here.

**LVL1 algorithms**

LVL1 algorithms must be simple to be realizable in hardware. The algorithms in the trigger simulation must correctly reflect the hardware. Several thresholds are allowed for each trigger object.

- **LVL1 muon algorithm**: selection of muon candidates using information from three layers of trigger chambers with two-coordinate read-out. The low-$p_T$ triggers require a coincidence between the two inner chamber layers, within a road that guarantees 90% efficiency for $p_T \sim 6$ GeV/c muons. For the high $p_T$-trigger (20 GeV/c) the third, outer chamber layer is included in the coincidence.

  LVL1 calorimeter algorithms: selection of (1) isolated electrons/gammas, (2) isolated hadrons, (3) jets and (4) measurement of missing energy.

- **For these algorithms the transverse energy $E_T$ is added up into trigger cells ($\Delta \eta \times \Delta \phi \sim 0.1 \times 0.1$) in the electromagnetic and hadronic calorimeters, and is converted to ADC counts. The algorithms operate on this grid of trigger cells. The optimal window size for adding up energy were found to be $2 \times 1 / 1 \times 2$ trigger cells for $e/\gamma$ triggers. For jets and missing $E_T$, the trigger cells are added up into coarser jet cells, typically of the size of $4 \times 4$ trigger cells. Sliding window algorithms find depositions above a set of thresholds. Isolation criteria can be imposed requiring that the energy surrounding the trigger object is below a given threshold.**

---

4 Software 44
Global LVL1 decision
The global LVL1 decision combines the LVL1 muon and calorimeter trigger objects. The trigger
decision is based on the multiplicity at each trigger threshold. More sophisticated decisions,
combining different trigger objects, like muon and jet, may be envisaged for future LVL1 trigger
menus.

LVL2 RoI guided algorithms
Each RoI accepted by LVL1 may be forwarded to the subsystem processors for the extraction of
physics signatures (feature extraction) in the muon system, the calorimeter system, and the in-
ner tracker (Silicon Tracker SCT and Transition Radiation Tracker TRT). The LVL2 trigger has to
access and process only a small fraction of the total detector data, i.e. the data in a region around
the RoI indicated by LVL1.

Each subsystem proceeds in two steps:

- preprocessing of the raw data in the selected region,
- feature extraction within the RoI.

Preprocessing consists of re-formatting and re-grouping the data. Careful tuning of the prepro-
cessing algorithms is as important as the tuning of the feature extraction algorithms.

The most important feature extraction algorithms at LVL2 are:

- Improvement of the muon track measurement using the high-precision chambers (MDT):
The pattern of hit strips in the muon trigger chambers is used to define a narrow road in
the precision chambers. Fast pattern recognition is then performed within this road, re-
sulting in 'super-hits' in the three muon chamber stations. Three superhits determine the
sagitta of the muon track. The sagitta in each \( \phi-\eta \) bin is related to \( p_T \); this relationship can
be pre-tabulated by Monte Carlo simulation.

- Identification of electromagnetic clusters in the calorimeters: The algorithm uses the
full-granularity, full-precision information from the calorimeter and preshower. The most
powerful parameters are: the transverse energy of the cluster, the lateral shape and longi-
tudinal shape that should be consistent with an isolated electron or photon, the isolation
of the EM cluster and the measurement of position in \( \phi \) and \( \eta \) for comparison with the po-

tion reconstructed from tracks. The profile of the energy depositions in the preshower are
used to enhance the \( e/\gamma \) signal compared to the signal from \( \pi^0 \)s.

- Track finding in the inner detector TRT and SCT: Hits are transformed into \( r-\phi \) projec-
tions in the barrel or \( z-\phi \) projections in the end-caps. Tracks are represented to a good approxi-
mation by straight lines in these projections, with slopes directly related to \( p_T \). The TRT
measures tracks in projection, whereas the SCT results are 3-dimensional tracks. In both
cases track candidates are found by fast histogramming methods that return approximate
\( p_T \) values. For the TRT these histogramming methods may be applied to coarse informa-
tion (straw positions) or refined information using drift time in addition. Fits are applied
to improve the momentum measurement. The algorithms assume constant magnetic
field. Track candidates are accepted if they fulfil

- track-finding criteria (number of hits relative to all crossed TRT-straws or SCT-strip
clusters)
- track-fitting criteria (\( p_T \), \( \chi^2 \), and criteria based on drift time)
- for electron tracks: electron identification criteria (matching between track and EM
cluster, and fraction of TRT-hits with transition radiation signals)
LVL2 algorithms without RoI guidance
For B-physics studies the LVL1 trigger will be an inclusive low-$p_T$ muon trigger. The LVL2 algorithm involves several steps: first the trigger muon is confirmed using the usual RoI-based algorithms; then a full scan over the whole volume of the TRT is executed and new RoIs are defined for tracks with energies as low as 1 GeV; finally the SCT, calorimeter, and muon data in the new RoIs are used to determine the track parameters in three dimensions and identify the tracks as electrons, muons, or hadrons for final selection using the B-physics trigger menu. Studies are under way to adapt the TRT algorithm, described above for high $p_T$ (> 20 GeV), to the lower $p_T$ region for use in the full TRT scan.

Global LVL2 algorithms
The global LVL2 algorithms are performed in two steps: first ‘trigger objects’ (gamma, electron, muon, hadron, jet, or missing-$E_T$ candidates) are formed, then the final LVL2 decision is made, based on the number of non-overlapping trigger objects found.

The trigger objects result from combining the features extracted in different subsystems:

- comparison of tracking results from the SCT and the TRT ($p_T$, $\phi$, $\eta$)
- comparison of tracking parameters and calorimeter cluster parameters ($E_T$ versus $p_T$, $\phi$, $\eta$)
- for muon candidates verification of isolation criteria
- correction of missing $E_T$ by accounting for muons.

The strategy for accepting an event is defined by trigger menus. Very loose selections are applied in the trigger simulation program to allow the study of the selection criteria; the selections are tightened only in subsequent analysis steps that work from NTuples that store the results of the subsystems and global processing.

4.2.5 Techniques

4.2.5.1 ATGEN, an interface to event generators

For the simulation of the physics processes in pp collisions, our software is interfaced to both parton shower type (PYTHIA, ISAJET, HERWIG) and matrix element type (NJETS, VECBOS) event generators in a way which makes it easy to change from one generator to the other without a detailed knowledge of how to use the different generators. Some of the features provided by the interface are listed below:

- The various generators are controlled through lists of statements which include a large number of options and parameters. A WWW interface assists in building the list of input statements while providing direct links to the generator documentation. A database of these lists of control statements is maintained to always be able to provide information on how a specific dataset was generated.
- There is a code management mechanism for the different generators in order to be able to reproduce previously simulated data.
- Output from the generators to be stored and passed through a full detector simulation.
- Simple parametrizations for detector responses are provided to allow for fast simulations at the particle level.
- At the particle level event data can be analysed with some generic analysis routines.
Future developments of the interface will be made to match the redesign of the ATLAS software into an object-oriented framework while at the same time following the progress on the event generators [4-3]. We maintain close contacts with related projects such as LHC++ (Section 4.6.4.4) and the GEANT4 (Section 4.6.4.1) to avoid duplication of effort.

4.2.5.2 The ATLAS GEometry description language AGE

Detector simulations using GEANT are an important component of the ATLAS software. To ensure a high level standardization and compatibility of the code written by different groups of users, the ATLAS detector simulation and reconstruction programs are written using a dedicated language, AGE [4-4]. This language is a FORTRAN extension developed and maintained by the ATLAS collaboration.

In order to be compiled the AGE source code is translated into FORTRAN by a preprocessor. A separate set of tools provides a formal definition of the AGE language and a corresponding parser to allow for multiple code generators to translate the AGE source code into other languages like C++ or HTML.

AGE includes several ‘GEANT operators’, supported by a dedicated GEANT interface library. Maintaining the GEANT specific tables of materials, volumes, hits descriptions, etc. and ensuring the internal consistency of most of the actual parameters of the GEANT routines, it significantly reduces the amount of information that the user should take care of and improves the robustness of the program.

To ensure standard data description and good control of data integrity, AGE includes data access operators, which together with C-like structure definition allows the control of data transfer between program modules.

Many important implementation details—such as memory management, data structure documentation, coding rule reinforcement etc. — are hidden from the user and may be updated by the software support team during the process of software development without requiring any changes in the code prepared by physicists.

This approach allows a higher level of data abstraction and a greater level of the code independence. It automates the detector response and data access mechanism, simplifies coding of the signal processing, and provides a data handling mechanism with built-in documentation and database support.

4.2.5.3 The ATLAS detector description in DICE

To describe the ATLAS detector in the detector simulation package GEANT, the user has to provide a set of ‘user routines’ for each (sub)detector to be simulated. For all subdetectors of ATLAS, these routines are maintained together in a package called DICE (Detector Integration Code). The code in DICE is written in a mixture of FORTRAN and AGE (for the geometry and digitizations). The compulsory user routines for each subdetector ‘DET’ are the following (always prefixed by the subdetector acronym ‘DET’):

- DETGEO which builds this geometry hierarchy and defines the materials
- DETDIG which defines the subdetector response to the traversing particle

Further optional user routines are:

- DETSTEP which defines the way the particle is tracked through the subdetector
• DETPRI which controls the level of printing
• DETHIS which defines histograms to be used
• DETUNP which defines the unpacking of the digitizations

The GEANT program describes any piece of the detector as part of a hierarchy of 'volumes'. The object-oriented version of this (GEANT4) will not follow this standard but rather use one from Computer Aided Engineering (CAE) packages which take planes as their basic unit. (However, GEANT4 will still know about volumes for reasons of backward compatibility). In the present ATLAS detector description any part of a subdetector or any piece of dead material like supports or read-out electronics and cables are described by volumes. To describe the whole ATLAS detector requires more than eleven million volumes. Because this level of detail is not always required for many investigations, parts of the detector can be switched on or off through data-cards. The main volume sets existing in DICE are:

• dead material
  • beam pipe
  • cryostats
  • toroid magnets
  • solenoid coil and flux return
  • inner detector and magnet supports

• inner detector
  • pixels (barrel and end-cap)
  • silicon (barrel and end-cap)
  • straw tracker

• calorimeters
  • electromagnetic accordion (barrel and end-cap)
  • hadronic barrel (tile)
  • hadronic liquid argon (end-cap)
  • very forward liquid argon

• muon spectrometer
  • muon drift tubes, mdt (barrel and end-cap)
  • resistive plate chambers, rpc (barrel)
  • thin gap chambers, tgc (end-cap)

The DICE package together with a control package called SLUG (I/O and GEANT interface) and GEANT are the central packages used for ATLAS simulation work. For the ATLAS event simulation for physics studies and for detector design, we use the shower packages available in GEANT3 (G-CALOR and G-FLUKA). A detailed discussion on the accuracy of shower packages can be found in the ATLAS calorimeter performance report [4-5]. The ATLAS requirements on the accuracy of shower simulation will be established and forwarded to the development teams in the near future. The above fairly complete detector description together with the graphics components of GEANT have been used for most of the published drawings. For the study of radiation levels in the ATLAS detectors, we have used a stand-alone version of FLUKA.
4.2.5.4 A C++ interface for the current software

A C++ interface to access the bank structure of the current ATLAS offline software has been developed. This development is to allow:

- mixing of C++ classes/objects with the FORTRAN routines of the current software, and
- uninitiated FORTRAN programmers to learn the basics of C++ while doing something useful/practical.

However, one must keep in mind that changing the current functional reconstruction code into C++ functional code is intended to allow people to begin playing with the language. This is not meant to replace the introduction of well-designed object-oriented code (see also Section 4.6).

The basic mechanics of the interface is:

1. One defines a data structure using the Structure definition of the AGE language (see Section 4.2.5.2) saving it in a header file.
2. Then one generates two interface classes for this structure using a code generator developed with a parser for the AGE language (see reference [4-4]).
3. Finally, C++ methods allow banks to be created/accessed given the path to the bank in the offline bank hierarchy, and C++ accessor methods return bank values by variable name.

This interface provides all of the functionality for data access of the current FORTRAN-based system, and will serve as a C++ training ground (see Section 4.6.1) for those interested while an independent object-oriented framework is being developed for the offline software (see Section 4.6.2).

4.3 Object-oriented software

We believe that the objectives for the ATLAS software, outlined in Section 4.1, can be best achieved by moving to object-oriented computing methods. The same conclusion has been reached by several other HEP projects, such as BaBar, CMS, RD41 (Moose), RD44, (GEANT4), ROOT (Analysis) and RD45 (Database for LHC). OO methods have been in use in online systems for some time.

Below, we contrast these methods with the traditional approach. To dispel a common misconception, the change is not primarily one of computer language; rather, we need to change our design structures, and the language change is then necessary to support the new design methods.

4.3.1 Traditional model for HEP software

The traditional method for producing HEP software uses a consolidated data-structure in memory, allocated to particular sub-tasks by a memory manager such as ZEBRA or BOS, and accessed by FORTRAN routines.
Using this method, it is almost impossible to achieve an adequate degree of encapsulation of different modules, signalled as essential in Section 4.1.2. Some improvement can be made by imposing data-access conventions and routines derived from the Entity-Relationship (E-R) model. This approach was implemented by ADAMO, in use in ALEPH, ZEUS and other experiments. Nevertheless, the concept of a single data-structure available to all modules is still preserved.

### 4.3.2 OO model for HEP software

The features of OO software are usually summarized as

**abstraction** identifying the essential features of an object, and presenting services offered by it as its public interface

**encapsulation** packaging the mechanisms supporting the services which the object offers, and the state variables used by those mechanisms, within the object itself

**data-hiding** keeping those mechanisms and variables invisible from outside the object, other than by the interface of services

**re-usability** ensuring that the services offered in an object interface are as general as is consistent with other requirements, so that they can be used in more than one part of the system

**inheritance** useful when a group of classes present a similar interface to the outside world but have different internal implementations of methods

**polymorphism** defining a service in the same way for two or more different types of object, even though the implementation of the service may be different, such that the object can provide the appropriate service without the caller being aware of which type the object is.

In this subsection, we interpret these rather formal terms in the context of HEP software, and indicate their individual importance for our work. As already stated in Section 4.1, the nature of the collaboration, with dispersed developers and a very long-lived project, leads to a particular emphasis.

For us, the main benefit is *encapsulation*. The different parts of the system are treated as *objects*, which include the data variables describing the *state* of the object, and the routines describing the *behaviour* of the object. How the object provides the services it offers, and what variables it uses to do so, need not concern the users (*clients*) of those services.

To take an example based on the reconstruction of an electron track, we need objects representing the electromagnetic calorimeter (ECAL), a track, a cluster, and so on. The state of the track includes variables for the components of the track's momentum. As an example of a service offered by the ECAL, one may ask it to return a list of clusters. The ECAL code and data are kept together, and the necessary design and implementation can be carried out and maintained by the ECAL domain team, with no interference to or from the other domains.

*Data-hiding* and *abstraction* are also implicit in what is written above. Encapsulated data not in the visible interface are hidden from the client, and so can be changed without penalty during development, provided that the advertised services are preserved. For example, different strategies and algorithms for track reconstruction can be introduced without affecting the user's perception of the system. Abstraction refers to abstracting the essential features of an object and presenting them as services, leaving the rest to the internal implementation. Such services can be invoked as queries or commands.
Inheritance is provided in OO implementation languages, and allows common features between objects of different types to be implemented once only, saving effort and increasing clarity through uniformity. Thus, if the ECAL and the hadron calorimeter (HCAL) have some common features, we treat them as specialized forms of a more general type calorimeter. Those common features are put in the class calorimeter, and the ECAL and HCAL classes inherit those features. Inheritance should only be used when there is a genuine conceptual similarity between the classes concerned, otherwise encapsulation is damaged. If in the above case the ECAL and HCAL inherit from calorimeter then it must be true that ECAL is a calorimeter.

Polymorphism means that two similar types of object can both offer the same service. That is, the client sees the same service, although the actions executed will be different. Polymorphism is often associated with inheritance; the service can be defined as virtual or deferred in the parent class (the class from which it is inherited), and implemented in different ways in the sub-classes.

4.4 The ATLAS software process

The need for a software process was identified in Section 4.1. For a project the size of ATLAS we must adopt appropriate engineering techniques for the construction of the software. While it is not right to overemphasize the analogy between software engineering and ‘real’ engineering, they do have some things in common, such as the importance of good design.

A software process can be thought of as a dataflow diagram with inputs being the requirements for the software system and the output being the software system itself—code and documentation. This single huge process is then split into other processes linked by dataflows which are the intermediate deliverables. If it were possible to keep the entire ATLAS software in one person’s head then it could be argued that this decomposition is unnecessary, however as soon as a second person wants to work on the software, the decomposition of the process becomes essential.

The principal deliverables introduced into the process are design documents. It is much easier to check that the design corresponds to the requirements for the software and that the code corresponds to the design than to compare code directly with requirements.

Many experiments have adopted some kind of software process but have not tried to formalize it. We choose to be more formal because of the size of the ATLAS project. The ATLAS Software Process (ASP) [4-6] sets out in some detail the order in which the various project activities take place and the input and output deliverables for each activity. By having all this defined in one document, those involved in the project know just what is expected of them, and can see how their work contributes to the whole system.

The format of many deliverables is carefully defined to encourage uniformity across the project. This will greatly simplify maintenance, it will allow procedures to be automated and will make life easier for all involved in the software, especially newcomers who want to contribute to the subsequent evolution of the system.

The ASP, though detailed, is not intended to add an unnecessary layer of bureaucracy to the development process, but it sets standards which should become a natural part of the life of an ATLAS software developer and which will result in a better software system which does what we want, with the minimum frustration for all involved.

Material for the ASP has mostly been adapted from books and courses according to the experience or prejudice of its authors. A part of the process is that it can itself evolve in the light of ex-
perience; and we expect to see many improvements to the ASP as currently defined. We shall be working with a new design paradigm and a new programming language, and do not yet have experience of how a large collaboration will adapt to this new environment. The ASP should not however be considered to be an optional extra. It may be wrong, in which case it must be changed.

To make it easier for people to get started with the ASP, two other documents have been written and will be maintained: an introduction to the ASP [4-7] and a tutorial [4-8].

4.4.1 Key features of the ASP

The ASP as defined in [4-6] is summarized here. As explained previously many details can be expected to change, especially at the beginning.

**Deliverables.** These are documents of various kinds, such as management plans, designs, or source code which developers produce at particular stages in the ASP. Their format and content are carefully defined in the ASP by sets of rules, some of which appear as appendices in [4-6]. Each deliverable is reviewed by a small group before being released for general use, so that there is a guarantee that it is of acceptable quality. The reviewers compare the new deliverable with the input document to that part of the process. For example does the design match the requirements? The deliverables are a set of related documents providing a complete, consistent, and up-to-date description of the software system, as it exists.

In addition there will also be a number of documents, such as preliminary designs or background papers on various technologies which will not have to conform to a particular format, or be subjected to a formal review. However, they must be included in the project’s overall documentation scheme.

**Evolutionary development.** One of the ways in which software is different from hardware is that it permits design changes in a module while the module is actually being developed. In many ways, this is a helpful feature, because there is no point in proceeding with an inadequate design if a better one is available. On the other hand, uncontrolled changes by anonymous programmers lead to a lot of subsequent difficulties with maintenance. The ASP addresses this problem by defining procedures to allow the system to be developed in controlled cycles.

After a preliminary phase, when some of the requirements have been collected and the outline of the system structure worked out, development is carried out in a sequence of short cycles (of eight working weeks, in our proposed scheme). At the end of each stage there is a new working system, though to start with, it will have very limited functionality. Each new cycle adds some more software modules, integrates them with the existing system and applies some global tests. This type of development is termed ‘evolutionary’ [4-9], and although not invented for OO development, it is highly suitable for it. Developers can work independently on implementing and testing classes during a cycle, and then incorporate them for system testing at the end of the cycle.

Even when the system has the planned functionality, we must expect that there will still be changes as algorithms are improved or the need for a new feature is identified. In many cases, the change will involve exchanging one version of a class method for another, or of plugging in a new class. With the good encapsulation that OO design encourages, these changes can be proposed for the next cycle and then introduced with little disturbance to the rest of the system and made public at the end of the next cycle.
**Project organization.** If the project is to run smoothly, responsibilities must be defined. Many people will have multiple roles in the software development.

The chief architect has the task of organizing a small group to define the overall structure of the software and dividing it up into domains corresponding to coherent work packages, some of which might be expected to correspond to the subdetector hardware. These are parcelled out to domain architects, who have the responsibility for organizing a domain team and designing and building the software in their individual domains.

Other roles include that of project manager who is responsible for setting objectives, planning the organization of the work, and allocating resources, and a number of more technical ones such as taking responsibility for project documentation or configuration management.

From a practical point of view, there is a need for hierarchy of organization, so that the individual domains (which might correspond to a well-defined unit of ATLAS like a calorimeter, or to a well-defined feature like a GUI) can get on with their own work semi-independently. However, there is also the need to ensure that the system will work as a whole. Accordingly, there is the concept of the core team, which once the project is rolling, consists of the project manager, the chief architect, the domain architects, and a requirements monitor to ensure that the software does not deviate from the needs of the users. It is this group which evaluates progress at the end of each development cycle.

**Quality assurance.** Two important mechanisms for ensuring and improving software quality are envisaged. The first has been briefly mentioned already: the review of deliverables. Starting from the document defining the user requirements, proceeding through various design documents to the final source code, it must be possible to show that each new deliverable satisfies the requirements implied by the preceding document in the sequence. In addition, for the sake of having uniform standards for structure, notation, and content for a given type of document, a deliverable must conform to rules defined in the ASP.

The second mechanism is longer term, and consists of identifying useful software metrics that might give an indication of quality. For example, classes with a very large number of methods, or very long methods, might be more bug-prone and more difficult to maintain. The metrics collector has the responsibility for collecting observational data, and determining what sort of values might correspond to 'large' or 'long'. As the cycles succeed each other, it should be possible to get an increasingly good idea of these values.

**Software process improvement.** As well as improving the quality of the software, we have the possibility of improving the actual process by which software is produced (as recommended in [4-10]). The metrics collector will also gather data on the time devoted to different kinds of activities and to the use of resources and may be able to identify ways in which the organization of the project could be improved. The fact that various aspects of the ASP itself are measured will provide a firm foundation on which to improve the ASP, though it is accepted that in the early stages changes without formal justification may have to be made.

### 4.4.2 Applying the ASP

We propose to apply the ASP for all of the ATLAS software development.

It is expected that the ASP will need modifications and it is part of the software process itself to modify the software process while getting more experience. We decided to start the ASP within the ATLAS software group on 1 September 1996 and are gaining experience. Some of the responsibilities described in the ASP document have been attributed to members of the collabora-
tion and some of the domains which came from an initial domain decomposition study for the reconstruction software have formed teams and are in the process of formulating requirements and producing designs.

4.5 The ATLAS software development environment

The Software Development Environment (SDE) is everything needed on the developer’s desk-
top (CASE tools, testing tools, compilers, linkers, debuggers etc.) in order to participate in the orderly development or modification of a software product. It should be a fully integrated operational software environment [4-11], and not just a collection of individual task-oriented tools. The users of the SDE are the developers of the final software to be produced within the environment. They need not be the end-users of the final software.

Although the software process is outside the scope of the SDE requirements document, its existence, definition, stability, and statistical control are necessary to allow a good identification of those requirements. As Stenning says “the role of an environment is to support effective use of an effective process” [4-12]. The software environment will be the instrumentation of the AT-
LAS Software Process (ASP) [4-6].

A working group has discussed the requirements and has produced a first document [4-13]. This document lists only the functional requirements and does not concern itself with issues of availability, pricing and licensing. It is expected that in two years most of the requirements will still be valid but a major revision of the document, based on the experience with the environment, is foreseen in phase with a re-assessment of the software itself. If it turns out that in some areas the requirements can not be satisfied, minor revisions may be necessary in the interim period. The document has identified a number of major development activities such as capture of user requirements and overall design. Orthogonal to these main activities are a number of sup-
port activities, such as configuration management and training which are needed for all or most of the main activities. The requirements were found by considering the support activities for each of the main activities.

To be able to proceed, some initial choices have been made for the Software Development Envi-
ronment. Although these choices may change with time for the two-year scope of this Comput-
ing Technical Proposal, it is worth mentioning them because it does give an idea of the envi-
ronment in which we propose to develop software. The full set of choices, with justification
(one of the deliverables of the ASP), will be subject of another document which will come out somewhat later than this CTP. The choices listed below concern primarily OO software develop-
ment.

The operational environment of the ATLAS Software Development Environment is mainly the development platforms but in some cases the requirements also concern the target platforms—where the software is to be run rather than developed. Today, UNIX is the main development platform, however many sites now promote PCs as the preferred desktop device. It has been de-
cided that all code must work on both Unix and on Windows NT.

4.5.1 Design methods

We have to decompose the global ATLAS software system into domains which can be designed
and implemented by separate teams. It is clearly very desirable that the different domain teams
agree on a common style of analysis and documentation for their designs, so that the domains
can be assembled painlessly into a total system. The documentation of the designs has a perman-
ent value for maintenance purposes.

Several design methods are available and mostly differ in the emphasis which is put on differ-
ent aspects of the analysis and the design process. Some of them have been studied by RD41
(Moose) and documents exist which compare methods and explain in which areas they are best
applicable. Recently the authors of the leading methods have been working together to define a
Unified Modelling Language (UML) which can be used by all design methods. It is likely that
UML will become the standard notation.

It has been decided that ATLAS will follow the Unified Notation as soon as the standard has
been published officially and tool support is acceptable. Up to that time the OMT notation for
diagrams showing static associations between classes (the Object Model in OMT terminology)
will be used along with a diagram which shows message flow using the notation supported by
the Object Interaction Editor of the StP CASE tool.

4.5.2 CASE tool

Computer Aided Software Engineering (CASE) tools can help in the design process. They pro-
vide the possibility to produce uniform designs for all parts of the software, and to document
these designs to make them more understandable. One can generate code (class headers) and
documentation from them or reverse-engineer existing code into these tools thereby providing
a permanent link between the design documents and the corresponding code.

The Software through Pictures (StP) CASE tool has been selected to support the design work.
Within this tool we will use the OMT notation but plan to switch to UML when it is available.
From the design in StP a design report can automatically be generated.

It is possible to store code fragments in StP so that full ‘generation’ of the code is possible. How-
ever it is expected that some developers may prefer to develop in code for a while and then ab-
stract their design back into StP. Another tool Sniff+ can be used alongside StP to reverse-engineer code into StP and so generate design reports corresponding to the imported
code. For this purpose some ATLAS-specific customization of StP has been developed.

It has been agreed that all code not developed through StP will be reverse-engineered into the
tool to maintain a coherent set of design reports for all the code.

4.5.3 Implementation languages

There is no realistic alternative at present to C++, in spite of its well-known deficiencies. Much
of the world-wide development in OO currently is based on C++, and support and tools of all
kinds are mostly only available for that language. It is easy to write bad C++, which is difficult
to maintain. The worst dangers may be avoided if we use a reliable tool to generate the C++
header files from the design.

We will continue actively to monitor developments in OO languages, and to make use of lan-
guages other than C++ if it appears advantageous. The design decisions should be language-in-
dependent.

Java is an example of a language which is becoming widely-used and has industrial support,
and so is worth further study. It is barely a year old, but already tools have been produced to
support it. It is a clean OO language with C-style syntax. It has built-in garbage collection, no
pointers to manipulate, and though it does not support templates, as the type of all objects is
known at run time, this is not a problem.

4.5.4 Compiler

It has been decided that all C++ code must compile with the gnu g++ compiler on Unix as well
as with the Microsoft Visual C++ compiler on PC.

Unfortunately some database vendors do not support the gnu compiler, but only the native
compiler. This would mean that we would have to require that all code can be compiled with
the native compilers from all platforms used within ATLAS. This may be a direction to follow
later in the project but now would lead to too much overhead as different vendors have imple-
mented different parts of the C++ standard.

4.5.5 Debugger

No decision has been made on what debugger to use on Unix. Very often people are used to the
native debugger on the platform of their choice. Microsoft Visual C++ under Windows NT in-
cludes as part of the environment an easy to use, though not very powerful debugger. A con-
ventional debugger is not very good for finding memory leaks and dangling pointers resulting
from inexpert use of C++. A tool, Purify, is able to track down bad memory management. It is
expensive; however, it has been decided that in case of difficulties, the Purify tool or a tool with
similar functionality must be available within the collaboration.

4.5.6 Programming environment

No decision has been made on the development environment for the writing and testing of code
on Unix. SNiFF+ is good and very fast and is needed for the reverse engineering into StP. How-
ever it is quite expensive. It is available under Windows NT, but it does not offer very much on
top of Microsoft Visual C++.

4.5.7 Project management

We propose to follow the lead given by ATLAS management and use Microsoft MS Project.

4.5.8 Verification and validation

Testing is vital at all stages of development. As testing is a rather boring job it is important that
whatever can be automated is automated. Tools can help with consistency checking of designs,
with generation of test cases, and with running those tests. Tools can also be used to test cover-
age of code by the tests and to look for memory leaks. These tools have not yet been selected.

It has been decided that all C++ code should follow a document defining standards [4-14]. This
is a first version of our C++ standards document so it is expected to change. These are not abso-
lute rules, and with justification they may be broken. We would like to use an automatic tool to
check code against the standards as far as possible. Various possibilities exist, but no decision
has been made yet. All such tools also allow numerous metrics to be generated from the code.
such metrics can be used to spot potentially poor quality code.
4.5.9 Configuration management

It has been decided to use cvs on Unix to manage the code repository. There will be a master repository at CERN made available to all members of the collaboration through afst. A working group has been set up to study the way in which cvs will be used (given that some people will now be using Windows NT) and how software releases will be carried out.

4.5.10 Documentation

No specific decision has been made on documentation other than that it will make the fullest use of the World-Wide Web. There is work going on within the LIGHT project [4-15] for automatic generation of the documentation on the web from C++ code, StP diagrams etc. for which a demonstrator will be finished by the end of 1996.

4.5.11 Training

We have used the CERN Education Service infrastructure to organize some basic software engineering training as well as more specialized training. We want to make this accessible to all members of the collaboration. A preliminary training plan has been discussed and is very close to what is offered by the service now (see Section 4.7).

4.5.12 Communication

It has been decided to advocate the use of MIME compliant mail user agents.

It has also been decided to promote video-conferencing (desktop or conference room) for all software developers within ATLAS.

4.6 Strategy for change

Particle physics has a long tradition in procedural programming using FORTRAN. A wide range of subroutines, and complete programs are available — for example those in CERNLIB. Moving from FORTRAN to OO will not be a trivial task if we wish to preserve the ideas embodied in this huge volume of code.

Probably the change of language is not the most important aspect although it is often felt that way. OO requires a different way of thinking about systems that people with just a FORTRAN background may take some time to grasp, though it should be less of a problem for the younger collaborators.

We also propose to change the procedure by which software is produced—an engineering approach for software development much the way the construction of the ATLAS detector itself is managed. It is generally accepted that one has to follow a well-defined procedure for the construction of a high-quality detector. We intend to follow similar procedures to develop high quality software.

The proposed changes will require a lot of new tools. We have not used many tools for the development of FORTRAN code and some fraction of those were home-made. With the wide acceptance of object technology in the software industry, many commercial tools have become
available to support software development and software process management. We plan to use the ones which will improve our software development.

To introduce OO and C++ we propose to follow two lines; one will start from the existing simulation and reconstruction code and make extensions and modifications in C++, and the other starts from scratch with a new OO design.

### 4.6.1 Benefiting from the existing FORTRAN code base

To guarantee that working programs for detector studies always exist, the current code will be maintained and improved until OO versions with equivalent functionality become available, a class interface has been developed (see section Section 4.2.5.4) and will be further improved to access the data produced by the existing simulation code. Using those classes the developer can now write reconstruction code in C++ and use data generated with the existing programs. Large chunks of code written in FORTRAN can be packaged together and wrapped in C++ classes (this will only work well if that FORTRAN package doesn't use data passing through COMMON blocks). This way the development of reconstruction code is little disturbed and it allows the developer to gain experience in using C++. In this approach it is possible to gradually translate existing FORTRAN into C++.

In spite of the advantages of this 'safe solution' there is the disadvantage of the compromise of the design. We will only profit from the advantages of OO if the code to be developed is well designed. If we start from a FORTRAN program we will not produce the best design because it will reflect the structure of the existing FORTRAN. So we propose a parallel line of development where we will still benefit from the algorithms and experiences reflected by the FORTRAN code but will start with a clean design.

### 4.6.2 Building a new software system from scratch

In this parallel development line a simulation and reconstruction package will be developed using OO starting from scratch. This effort will need some part of the time of those working on the existing software and some new people. The overlap of the groups is important to preserve the knowledge which was acquired in the development of the existing programs.

It is felt that pursuing these two parallel lines of development minimizes the risk of the changeover from procedural to object-oriented programming. It does not disturb significantly the necessary detector studies based on simulation. It allows the development of new FORTRAN code which follows closely the development of ideas on the ATLAS detector itself. The C++ for the current software interface (see Section 4.2.5.4) offers an easy migration path for those who want to start working on the new code. The second development line guarantees that a proper basic design will be done as well. It might be hoped that it will be relatively easy at some point in time to use large chunks from one system into the other. Also, this dual development makes irreversible decisions for the one or the other paradigm less time-critical. The danger of this approach of making it easy to develop code for the one or the other system is that the need for the changeover will not be felt strongly enough so that not enough people will join the design and C++ coding efforts.
4.6.3 ARVE\textsuperscript{1}, an analysis framework design

ARVE is a prototype object-oriented framework for reconstruction and physics analysis which makes it easy to test design and implementation ideas from the approach described in Section 4.6.2. As it stands it is incomplete but not far from something which could be used for (sub)detector performance studies. It must be noted that the ARVE development went faster than the evolution of our ideas on the software process. At this moment we are correcting this by producing design documents of the various aspects of ARVE.

The ARVE framework has been developed with the needs of two rather different users in mind: the developer, who wants to be able to rapidly test code in the context of the final application, and the user, who needs an intuitive, familiar environment, with interactive and batch capability.

In other words, the development environment must be a subset of, and easily evolve into, a production environment. To satisfy this, we have developed a very modular and flexible scheme. The modularity is primarily with regard to how easy it is to create applications containing only the subdetectors relevant to a developer. The control model follows this scheme, in that subdetectors are responsible for defining their commands, via a common interface.

Rapid testing and development is made much easier with another capability that we provide: the ability to generate and simulate simple events, perhaps single particles, with parameters that can be easily modified.

All this is exemplified by Figure 4-1. It is a conventional window, containing a title bar, menu bar, and graphics display area. A PC running Windows NT was used to generate the display, but the graphical user interface definition allows other implementations: the appearance of the X-Motif version is identical.

The display is of a projective r-\(\phi\) view of the ATLAS detector, with some of the barrel muon chambers, and the solenoid shown. Also shown is a single 20-GeV muon, (under control of an object that responds to commands from the 'Test Beam' menu) which bends in the central field, then intersects three muon stations. The marks where the track intersects the chambers are generated independently, by objects that record hits. Although it is a projective, 2-d display, a perspective view is possible, along with rotations, zoom, and pan. Finally, it is possible to generate a file of VRML commands that can be viewed with a 'virtual reality' browser.

It should be emphasized that this graphical ability is intrinsic to the design: it is not a separate 'event display' package. As such, the user/developer can easily implement display of particular objects corresponding to response of detector elements or to the reconstruction. But it lacks some of the special views and other capabilities associated with event display packages, such as ALEPH's DALI\textsuperscript{2}. An independent effort is under way to define event display requirements for ATLAS.

4.6.3.1 The domains

The elements in the display of Figure 4-1 all have associated objects. These are allocated to domains, a partitioning of the problem into distinct pieces to aid in the design, and allocation of effort. An initial list of top-level domains associated with the framework follows:

\footnotesize
1. Acronym is ATLAS Reconstruction Visual Environment.
**Subsystems and the ARVE class**

The ATLAS detector naturally divides into different physical subsystems, for example central tracking, muon barrel, etc. Each is geometrically distinct, and mostly autonomous. However, all must respond to requests to accumulate data (either from an event source, or generated on the fly), and especially to reconstruct. During reconstruction, moreover, several systems may need to cooperate. Here is a need for subsystems not connected with specific detectors, namely tracking that combines information from different systems.

The Arve class instantiates the singletons **GUI**, **PrintControl**, and **DisplayControl**, (all described below), and has a list of **SubSystem** objects. The method **attach**, adds to the list. Finally, **run()** is passed to the singleton **Menu**.

**Control**

All programs designed to be interactive require a means for the user to be aware of, and specify options. This need is fulfilled by the class **Menu**. Also, all such programs require services, in the form of print and perhaps graphics. We have encapsulated this idea as well, using the classes **Server** and **Services**. Our model of a program is that it is composed of a list of **Servers**, which need to be instantiated, then configured for a particular application. Then the program is ‘run’, meaning that a list of commands is processed, or user input begins. Before this happens, some servers may require notification to finish set-up.
Geometry
The geometry domain contains the properties of lines, surfaces, and volumes, and especially tracking in the form of determining the intersections of trajectories with surfaces.

Graphics
This portion of the ATLAS reconstruction and analysis prototype describes the mechanism for objects to present 3D information to a rendering system, usually, but not exclusively, as wire-frame models.

Simulation
A key part of the framework for reconstruction is the possibility to generate simple events on the fly. Thus we identify simulation as a domain, and expect that the full functionality of the standard simulation package can be integrated into the framework. This will eventually be the GEANT4 package, but until it is available, we use parts of an older package, Gismo\(^1\). Simulation can be decomposed into several subdomains:

- particle properties: mass, charge, width, etc., and decays to form a tree structure
- particle propagation: correlation of momentum and space, with effect of interactions in matter
- material descriptions
- particle interactions in matter
- digitization

All this of course depends on the domain geometry.

4.6.4 Benefiting from other object-oriented software projects
There are a number of relevant OO projects going on at the moment. These may bring us useful classes or at least ideas.

4.6.4.1 GEANT4 (RD44)

GEANT4 is the OO version of the GEANT detector simulation package. The project to re-engineer the existing GEANT package written in FORTRAN started in 1994. It is expected that a first version of a usable toolbox will become available during 1997. It is the plan to have some of the people who are both part of the GEANT4 project as well as of the ATLAS collaboration work on an early implementation of the ATLAS detector using the new GEANT4 geometry classes. The simple geometry package which we are using in ARVE as mentioned in Section 4.6.2 will then gradually be replaced by the corresponding package from the GEANT4 toolbox.

We hope that we will not have two independent packages, one for simulation and one for reconstruction. We shall collaborate with the GEANT4 team to achieve this if it seems reasonable.

4.6.4.2 ODBMS (RD45)
The RD45 research project studies the possibility to use a commercial OO Data Base Management System (ODBMS) for persistency management. ATLAS must decide how it wants to use

\(^1\) http://www.phys.washington.edu/~burnett/gismo/
the ODBMS and which objects should be persistent. If we use the ODBMS as it was designed to be used, we may become too dependent on the database vendor and at some point we may have to compromise our design because of inefficiency in dealing with very small objects. If we only access it through a special interface, rather than directly, this will make us independent of the database, but we may lose many of the advantages of the ODBMS. We need more experience before deciding which way to go.

4.6.4.3 ROOT

The ROOT system is the OO equivalent of PAW and PIAF: an OO framework for physics data analysis. It has been developed within the NA49 collaboration and has been very successful for the analysis of the many complicated heavy-ion events from that experiment. ROOT offers its own I/O scheme which is quite different from that described in Section 4.6.4.2. and histogram classes. It is free and is still being developed. The class hierarchy may not be very convenient for us so we may profit more from the ideas than from the code.

4.6.4.4 LHC++

LHC++ is a working group which is beginning to understand how something roughly equivalent to the current CERNLIB might be provided for future HEP experiments. The primary focus is on the needs of LHC experiments, and on C++ based solutions. LHC++ has both HEP-specific and commercial components.

One of the HEP-specific components is GEANT4 which was described in Section 4.6.4.1, and another is CLHEP, a C++ class library for HEP which was started during the Annecy CHEP conference in 1993.

The LHC++ working group is also looking into commercially available class libraries. A common selection of class libraries within HEP will save manpower and money. Products being considered include, Objectivity an ODBMS which is described in Section 4.6.4.2, IRIS Explorer, a visualization system, mathematical libraries and Rogue Wave's tools.h++.

The standard template library (STL) is becoming a part of the C++ standard (as the C++ standard library). This makes a number of classes in the existing CLHEP and most of tools.h++ unnecessary. We plan to make full use of STL and, like the LHC++ working group, consider that tools.h++ is only an interim solution. In cases where there is a choice we will use STL and avoid tools.h++.

4.6.4.5 BaBar

BaBar, which will take its first data in 1998, is the first big experiment which will rely on OO software for its simulation, reconstruction and analysis. It therefore is an important testbed. Collaborators from BaBar participate in many of the projects mentioned above and there are many informal contacts between BaBar and ATLAS people.

4.7 Training

The requirements for training have been formulated in [4-13]. We propose to formulate a training plan which fulfils those requirements.
Training must be organized now for software developers of the general packages who have experience in procedural programming but want to start using OO techniques and for the ATLAS physicists who will be working with those packages. However, this training has to be available during the whole lifetime of the experiment because there will be many new people who want to participate.

The first attempt to define a set of courses for ATLAS corresponds very much to what is offered in the technical training programme of the CERN Education Service. All ATLAS collaborators must have access to these courses. In case that it is more economical to organize training at some collaborating institute rather than having the people from such an institute come to CERN, personnel from the CERN Education Service will be asked to advise and assist in making the necessary contacts. A first version of a training plan has been evaluated in the past with members of the CERN training committee for software development tools. Thus the ATLAS training plan and their program at this moment are very similar (see Figure 4-2).
A basic introduction to software engineering should be followed by all members of the collaboration who want to do any computing. A second general introductory course 'C++ for particle physicists' is strongly recommended to introduce some of the ideas of OO and to teach the first principles of writing code in C++.

Object-oriented Analysis and Design (OOA&D) and Object-Oriented Programming (OOP) training is not to be followed by the same number of people but is necessary for the software developers of the more general packages. In the OOP training C++ or Java will be used and the courses should use examples from particle physics. An advanced course in C++ should exist for developers who have been using the language for some time. Topical training may be organized, which might then become part of the standard training package, on special occasions such as the introduction of the Unified Modelling Language.

Apart from this training in OO methods and languages, specific training has to exist for all the components of the Software Development Environment. This may range from class room courses for OO Database Management Systems (Objectivity course) or CASE tools (StP course) which are part of the standard training package to tutorials or videos for the somewhat simpler tools we will use.

Each tool or technique used in the ATLAS Software Development Environment has one name connected to it of a person who is responsible for the definition of the appropriate training for that component. It is also his/her task to organize consultancy if and when needed. Again the same person is responsible for a tutorial with some ATLAS-specific examples.

A training plan is one of the deliverables of the ASP and will be a separate document. In the review it will be compared to the requirements listed in [4-13]. This plan will change in time, but what is important is that we always have an up-to-date plan so that people with any background can always be trained to join the software development team.

### 4.8 Planning

The development will proceed in small cycles of eight working weeks as defined in the ASP. At the end of each cycle a one-week workshop is organized where progress is discussed and where the plan for the next eight weeks is made. At the end of each cycle a new working version of the software is made. Four such cycles will fit in one year.

A more global plan is made on the basis of two years where the major milestones are defined. Moreover two years is a reasonable timescale to review important decisions such as method, language, and CASE tool. A shorter timescale would not give sufficient stability to make progress. This two-year cycle also coincides with the upgrade frequency of this Computing Technical Proposal.

The global plan is:

- **1997-1998 initial version 0**

  Existing code in FORTRAN and ZEBRA is further developed and modified and interface classes are built to allow users to access all data structures in C++.

  Requirements for the overall ATLAS software are formulated and a global decomposition of the domain is made where subdomains are identified. A framework of subdomains is built for detector simulation and reconstruction. All subsystems of the ATLAS detector
are included but with limited functionality. Requirements and a design are worked out in more detail for each subdomain and code written.

The framework itself will have possibilities for single-particle simulation and tracking but for detector studies data will be used generated with GEANT3.

Studies will be made on how to use the classes from the newly developed GEANT4 toolbox and how persistency can be organized using Objectivity.

- 1999-2000 version 1
  The FORTAN and ZEBRA code will be maintained and further developed but by the end of this cycle the OO version of the code should be usable for detector studies.
  This means that the ATLAS detector will have been completely described in the GEANT4 framework. The database should now take care of the persistent objects.

- 2001-2002 version 2
  The framework will have the full functionality to do event and detector studies.
  Distributed computing will be studied because the parameters of the computing model should be better known by then.

- 2003-2004 version 3
  The framework is integrated with the data acquisition and trigger systems and performance issues are studied.

- 2005 production version 4
  Production version

For the more detailed planning of the software development one should consult the eight-week cycle plans which will be maintained on the web.

4.9 Costs

In Section 4.5 a first list of commercial products was presented which are part of the ATLAS Software Development Environment. More tools and libraries may be needed so it is still difficult to estimate exactly what the total costs will be for software development. The costs will vary per country and will depend on the type of participation in software development and will also change with time. We intend to minimize those costs in order to keep the threshold for participation as low as possible. However, the choice to use commercial software tools will make us pay some money.

In the context of the CERN LHC++ initiative some of the software tools will be bundled together and HEP wide runtime licences will be obtained. Developer licences should be obtained as needed, although under HEP-wide conditions. The various components are listed separately below.

Database software: Objectivity
There is a difference in price between developer licences and runtime licences but very few (order 10 to 20) people will need developer licences. At this moment there are sufficient developer licences (order 10) for ATLAS people at CERN. Developer licences for the institutes will have to be paid for by the institutes but CERN will assist in finding the local dealer and getting ‘CERN’ prices. Only those people who are involved in designing and building the object model for the persistent objects need such a developer licence. Most developers and users who want to use
the objects or want to create new selections and keep them on the database can do so with runtime licences. It is expected that 250 runtime licences will be sufficient for the whole collaboration. Some of the larger institutes already have their own developer licences but institutes who don’t and who do want to participate in this sort of development will have to pay unless the development can be done remotely using one of the seats at CERN.

**Graphics software: Iris Explorer, Open Inventor and OpenGL**

The Iris Explorer graphics package is built on Open Inventor and Open GL and only in exceptional cases will people need to pay additional money for developments which aren’t covered by the Explorer licence. The Iris Explorer CERN site licence should be extended to all external sites associated with CERN. This will increase the price but would allow all ATLAS members to use this package for their developments at an almost equal level. It is likely that there are more people who want to work on graphics developments than there are people developing the persistent object model so there is more reason for a wider licence agreement than for the database.

**General support libraries**

It is difficult to predict even on a short time-scale what will happen with libraries, but it is clear that we will need some general-purpose and some mathematical libraries now and in the future. The Rogue Wave tools.h++ is a very popular library and is used extensively in for example GEANT4. However, it is likely that a lot of its functionality is also provided by the Standard Template Library STL which is now part of the C++ language. There are good reasons for trying to minimize the usage of external libraries in our code and that should be our final goal. For the near future, however, we have to be able to use and test those libraries and a moderate number of user licences have to be available. We expect CERN to continue the support for those libraries in LHC++ and to make sufficient licences available so they can be used and tested.

It is expected that the necessary licences for the packages in LHC++ will amount to 50 kCHF per year for the whole of the ATLAS collaboration. This includes all required developer licences at CERN and all run-time licences for the whole collaboration. It does not include developer licences needed at the institutes.

Some other software tools which are not in LHC++ are needed for software development as described in Section 4.5. It depends to which type of software development one wants to contribute but in principle everybody in the collaboration should have easy and affordable access to any of them.

**CASE tool: StP**

The Software through Pictures, StP CASE tool is used within ATLAS for the design of our software and will be used for the generation of many of the documents needed in the software development process. At the moment CERN has made 15 floating licences available which can be used by anyone. This has been sufficient at this early stage but will not be enough when more people participate in the design and code development. We expect that no more than 30 licences will be needed for the next two to three years. It would not be wise to get a licence agreement for much longer than that because this market evolves very rapidly. We expect CERN will continue to coordinate the availability for this tool which may cost a total of 50 KCHF to the collaboration over the next three years.

**Code development environment**

There is a large difference between the cost for this type of software on Windows and on Unix. On Windows packages like the Microsoft Visual or Borland environment cost a few hundred CHF and give a very complete and user-friendly access to most functionality needed for code development. On Unix only products like SNIFF+ or Objectcenter have similar functionality but cost much more. One can of course develop code on Unix without those tools but it will not be
as smooth and user friendly. We expect that with the growing popularity of PCs people will migrate to Windows products to develop code.

To be able to reverse-engineer code which has been developed on Windows into StP to make for the generation of the design and code description documents, SNiFF+ will be needed again but not for every developer. A scheme, making use of the Web or remote access could be developed so anyone developing code can have access to this functionality. The same holds for code checkers, the tool which verifies that the written code conforms to the code guidelines. These tools would be fairly expensive (some 10 kCHF) but could be made available to anyone through some sort of remote access scheme since they will only be used occasionally. A few more tools fall into this class and we expect them to be made available for all software developers within ATLAS without the need to buy them at all participating institutes.

4.9.1 General support tools

Almost anyone using a PC will also want to have MS Office part of which is Excel which is the ATLAS standard spreadsheet package. Very often this software is nowadays part of the package deal when purchasing the PC hardware. MS Project is not part of Office but has to be purchased separately. Every institute will probably have its own channel to get those products for some discount price (order 1 kCHF) but we would encourage CERN to provide such ‘standard tools’ through, for example, the User Consultancy Office, which already has a similar service for books which is highly appreciated.
4.10 References

4-1   ATLAS Technical Proposal, CERN/LHCC/94-43, LHCC/P2, 15 December 1994

4-2   Trigger Algorithms in ATRIG Version 1.30, ATLAS
      Trigger Performance Working Group, ATLAS Internal Note, DAQ-NO-60

4-3   Proceedings of the Fourth International Workshop on Software Engineering, Artificial
      Intelligence and Expert Systems for High Energy and Nuclear Physics, Pisa, Italy, World
      Scientific 1995

4-4   Christian Arnault, AGE Version v1, ATLAS Internal Note SOFT-NO-20, (also available from

4-5   ATLAS Calorimeter Performance, CERN/LHCC 96-40, 1996

4-6   S.M. Fisher, K. Bos and R. Candlin, The ATLAS Software Process, ATLAS Internal Note
      SOFT-NO-02, (also available from
or as postscript in

4-7   Rosemary Candlin, An Introduction to the ATLAS Software Process, ATLAS Internal Note
      SOFT-NO-026, (also available from
html or as postscript in

4-8   R. Candlin, Experience with the ATLAS Software Process, ATLAS-OO Note 14, (available from


4-10  W.S. Humphrey, Managing the Software Process, Addison-Wesley, 1989

4-11  W.S. Humphrey, CASE planning and Software Process, CMU Technical Report,
      CMU-SEI-89-TR-26, May 1989, (also available from
http://www.sei.cmu.edu/products/publications/89.reports/89.tr.026.html)

4-12  V. Stenning, On the Role of an Environment, 9th International Conference on Software
      Engineering, Monterey, California, March 30, 1987

4-13  ATLAS Software Development Environment User Requirements Document, ATLAS Internal
      Note SOFT-NO-34, also available from
http://atlasinfo.cern.ch/Atlas/GROUPS/SOFTWARE/NOTES/note34/asde.html
or as postscript in
http://atlasinfo.cern.ch/Atlas/GROUPS/SOFTWARE/NOTES/note34/asde.ps

4-14  S.M. Fisher and L. Tuura, C++ coding standards for ATLAS, ATLAS Internal Note
      SOFT-NO-029, also available from
or as postscript in

4-15  Luciano Barone, Bertrand Rousseau, LIGHT/OO: A Prototype for OMT and C++,
      http://www.cern.ch/Light/coolight.html
5 Trigger and DAQ

5.1 Introduction

In this chapter we address computing and software issues related to the ATLAS Trigger and Data Acquisition (T/DAQ) sub-systems. By their nature the T/DAQ sub-systems encompass a wide spectrum of computing and software: hardware-based components, such as the LVL1 trigger; highly demanding real-time applications, such as the LVL2 trigger and the DAQ data flow management; as well as applications in some respect closer to "offline" computing, such as the Event Filter\(^1\).

Given the scope of the present document and the time scale of the R&D programme in the T/DAQ areas, we will mainly focus on the Event Filter as far as computing is concerned. As regards software issues, we will widen the scope also to real-time applications by complementing the software environment described elsewhere in this report with requirements and considerations typically related to DAQ.

The Trigger and DAQ sub-systems are due to present a Technical Proposal in December 1997, this document will deal with financing and sharing of responsibilities. Such issues are therefore not discussed in the present document for the T/DAQ sub-systems.

5.2 T/DAQ Computing

5.2.1 Working principles

The goal of the ATLAS T/DAQ sub-systems is that of 1) reducing the event data produced at 40 MHz rate to the current nominal target of a manageable 100 MB/s to permanent storage and 2) moving and distributing the correspondingly huge volume of data from the detector front-end electronics to the permanent storage [5-1].

Rate and data reduction

Rate and data reduction is organized in ATLAS in three trigger levels (LVL1, LVL2, Event Filter), as shown in Figure 5-1. At LVL1, special-purpose processors act on reduced-granularity data from a subset of the detectors. The LVL2 trigger uses full-granularity, full-precision data from most of the detectors, but examines only regions of the detector identified by LVL1 as containing interesting information. At the level of the Event Filter, the full event data are used to make the final selection of events to be recorded for offline analysis.

The LVL1 trigger accepts data at the full LHC bunch-crossing rate of 40 MHz (every 25 ns). The latency (time taken to form and distribute the LVL1 trigger decision) is ~2 microseconds, and the maximum output rate is limited to 100 KHz by the capabilities of the subdetector read-out systems and the LVL2 trigger. At high luminosity (10\(^{34}\) cm\(^{-2}\) s\(^{-1}\)), each bunch crossing contains an average of about 24 proton-proton collisions. Hence, the LVL1 trigger must select no more than one interaction in ~10\(^4\) (one bunch crossing in 400).

---

1. In other parts of this document the Event Filter is also referred to as Level-3 trigger.
The LVL2 trigger must reduce the rate from up to 100 kHz\(^1\) after LVL1 to \(~1\) kHz. The LVL2 architecture is based on the use of Regions of Interest\(^2\) (ROIs). The LVL2 trigger then has to access and process only a small fraction of the total detector data, with corresponding advantages in terms of the required processing power and data-movement capacity. The total LVL2 latency is variable, up to \(~10\) ms on average.

After an event is accepted by the LVL2 trigger, the full data are sent to the Event Filter (via the event builder). Complete event reconstruction is possible, with average decision times up to \(~1\) s. The Event Filter system must achieve a data-storage rate of \(~100\) MB/s by reducing the event rate and/or the event size. For some triggers, for example Higgs boson candidates, the full event data will be recorded with an event size of \(~1\) MB, allowing a maximum event rate of \(~100\) Hz.

**Data Acquisition (DAQ)**

When an event is accepted by LVL1, the data from all the front-end electronics systems are transferred to off-detector read-out cards, containing readout buffers, (ROBs). Only data required by the LVL2 algorithms are accessed by the LVL2 trigger system from the ROBs. Most of the data are not accessed from the buffers during the LVL2 latency.

The full detector data for each event accepted by LVL2 are transferred from the ROBs via the event builder to the Event Filter system for the final stage of selection before recording.

Data have to be distributed, from any level in the system (ROBs, crates, full subdetectors, farms), to analysis programs making asynchronous requests for detector monitoring and data

---

1. Although 100 kHz is the design objective, the system will initially work at 75 kHz.
2. The LVL1 trigger system is used to identify the regions of the detector containing interesting features, such as high \(p_T\) e.m. clusters, jets and muons.
quality checking. Detector calibration, in the form of a special run or of special events intermixed with the 'normal' data, also has to be supported by the DAQ.

Fundamental to the coherent working of the triggers and the DAQ data flow is the "back-end" DAQ component, encompassing all the non-time-critical/non-data-flow oriented DAQ functions. The run control, configuration databases, user interface and other ancillary applications belong to the back-end DAQ.

5.2.2 The LVL1 and LVL2 triggers

The ATLAS LVL1 trigger system is fully based on hardware, therefore it does not raise any computing or software issues (other than the general ones of control, monitoring, etc.). The LVL2 R&D and demonstrator programmes are investigating the relative advantages and applicability of commercial processors and networks, FPGA-based processors and dedicated data collection systems, as well as hybrid solutions, to the special problems presented by a high event rate and short processing time. In addition, the event analysis and selection algorithms at LVL2 have to use simple, fast algorithms and optimized code. Results from these investigations are expected for the T/DAQ Technical Proposal due at the end of 1997.

For the above reasons both LVL1 and LVL2 are considered beyond the scope of the present document.

5.2.3 The event filter

The Event Filter computing instrument will be provided by a processor farm. In the present view of the ATLAS DAQ, data coming via the Switch Farm Interface (SFI) will be distributed to several processors, working in parallel, each one analysing a full event. The farm should provide enough processing power to build the high-level information which will allow the Event Filter to take the decision of accepting or rejecting the event. This implies the capability of reducing the accepted event rate by a factor of 10 after the LVL2 system, bringing it to ~100 Hz, working with built events of 1 MB size. The farm should, moreover, have the appropriate bandwidth for the permanent data recording, estimated now at 100 MB/s.

Further requirements might come from the necessity of accepting and processing different trigger types, including physics events, calibration and monitoring data. In this context special care must be taken to avoid the creation of bottlenecks due to lack of resources while retaining the highest possible flexibility.

It is recognized that the final performance of this system will be determined by the maximum manageable complexity and the available financial resources. Extrapolations based on the reconstruction of CDF/D0 events and of ATLAS Monte Carlo events provide an estimate of approximately 25 SPECint95-sec per event. This brings the total amount of CPU required by the Event Filter process (excluding calibration and monitoring) to 2.5x10^4 SPECint95.

Our aim is that the filtering program which will run on the farm will be as much as possible the same as the offline reconstruction one, and no development of specific Event Filter algorithms is foreseen (unlike the case of LVL2). The software will be adapted to the particular needs of the real-time DAQ environment, ensuring in the first place reliability and robustness. This may be achieved by using special paths inside the reconstruction program. The code for calibration and monitoring will be of a different nature and more closely linked to specific requirements of the subdetector groups; running these programs might imply a more complex partitioning of the farm itself.
Depending on the time frame in which reliable calibration will become available, the possibility may be considered to output some reconstruction results coming from the Event Filter, together with the raw data, to help the work of the offline program.

5.2.4 The detector control system

The Detector Control System (DCS) is neither directly connected with physics event records nor with the flow of these data and hence it is not treated in this document. Its role is described in the ATLAS Technical Proposal [5-1]. DCS will, however, be fully integrated in the ATLAS computing environment and will make full use of the infrastructure as described in the present document.

5.3 T/DAQ software

The T/DAQ architecture highlighted in Section 5.2.1 provides programmable processing power at many levels - in the ROBs, DAQ crates, LVL2 trigger, Event Filter and back-end workstations. Although each level has specific requirements, a uniform environment, capable of providing most of the features required, is desirable.

5.3.1 Application software

General design principles
The software system must be designed to support the hardware architecture features of modularity, scalability, and openness.

Modularity means that the software is decomposed into independent modules, each providing a well-defined function. Modules can only be accessed via their implementation-independent interface. Thus, the risk of compromising overall system integrity when replacing components is minimized.

Scalability means that the software architecture will be based on scalable concepts such as the data-flow mechanism and distributed computing support. The configuration of the software architecture will be organized with databases, thereby providing a high potential for scalability.

Openness and platform independence require that the software should be based on industry standards. The integration of additional or different hardware and software is therefore potentially easier.

Software components
At the moment, it is neither possible nor desirable to produce a detailed design of the software that supports the Trigger and DAQ applications. However, a broad outline of aspects of the software can be given, based on extensive previous [5-2] and ongoing [5-3] R&D and prototype work.

The principal role of the software system is the correct execution, control and monitoring of event triggering and data flow. The major application areas are discussed in the following paragraphs.

- **Data-flow management.** Data flow management is the framework supporting the orderly flow of event data from the DAQ crates, through the event builder and Event Filter systems, to permanent storage. This is an area where DAQ hardware and software will be
very tightly coupled and which has the most demanding real-time requirements (viz. the 100 kHz input rate). It is safe to predict, on the basis of R&D activities such as those documented in Ref. [5-5], that evolution in DAQ hardware will help in migrating a sizeable part of this function directly into hardware.

- **Distributed computing support.** The DAQ system will, to a large degree, be decentralized, with functions distributed over several processors. While the data-flow mechanism supports the flow of experimental data (a large volume of data at high rate), the distributed-computing tools support the flow of control and monitoring information (a small volume of data at relatively low rate). There is considerable prototyping activity in this area; for example see Reference [5-3]. The use of commercial solutions is anticipated.

- **Databases.** Databases of various kinds are pervasive in the DAQ system (configuration description, run parameters, etc.). A common framework to design, maintain, and operate databases is essential. The databases must provide access to data at run time with an acceptable latency and be visible concurrently to client applications. Solutions which are common throughout the experiment, or even across LHC experiments, are an obvious goal. This is another area of considerable R&D [5-4] and prototyping activity [5-3]. There is confidence that a commercial solution will be available.

- **Monitoring.** A system to monitor the data quality and the performance of the DAQ is required. A framework for the control, monitoring, test and calibration will also be needed. The customization of commercial products to derive integrated solutions has been successfully evaluated [5-2], and this trend is expected to continue.

- **DAQ controls.** The control of the DAQ system is a complex problem involving distributed computing, databases, user interfaces, and decision-making software. Both the integration of various commercial products into a custom application and the use of specific industrial solutions have been evaluated [5-2] and prototyping work is ongoing [5-3]. Again a path for the use of commercial products is emerging.

It is anticipated that commercial software will be pervasive in the T/DAQ software system. Emphasis will be put on the integration of products into customized applications. It is recognized that a few areas may need fully customized solutions if they are specialized or demanding in performance.

### 5.3.2 Software environment

**Operating system**

It is desirable to have a common operating system (OS) across all levels of T/DAQ computing, possibly common also with the offline; prototyping work is addressing this issue. The OS environment must have features ranging from fast real-time response to support for running 'general-purpose' tasks (run-control applications, monitoring tasks, etc.). The real-time environment, therefore, requires a compromise between ultimate real-time performance (e.g. an upper limit on the time to respond to external events) and standardization (i.e. platform independence) in a system which supports distributed processing.

Requirements of T/DAQ real-time applications include the following: fast response to external interrupts and easy access to the external I/O resources; efficient inter-process communication and management of multiple event sources in an asynchronous way; ability to drive and/or monitor specialized hardware.

Performance is a major consideration in real-time computing. An important aspect of this is how long an external event (i.e. interrupt) can be delayed because of other activities in the sys-
tem, without stopping the system from working. Different levels in the system have varying dependence on this aspect. However, the following are generally required: low overhead for interrupt servicing, process scheduling, context switching and inter-process communication, and also efficient extension to a multiprocessor environment.

Today, real-time UNIX systems seem to be the best candidates for achieving a high level of uniformity and compliance in OS standards with real-time capabilities. Current research in ATLAS [5-3] is based on UNIX, but developments in standardization and real-time features will be tracked. The emergence of the WidowsNT operating system is also an issue which is being addressed in the DAQ environment [5-3].

The most suitable operating system commercially available will be adopted. Whichever OS is used, it must be as independent as possible of hardware architecture since, in order to profit from hardware developments and market competition, it must be possible to migrate the DAQ software to new architectures as they become available.

Development environment
Processors at many levels of the systems will run a full multi-tasking OS and support DAQ components for the readout and control of the local detectors, data recording, and online monitoring. Full access will be provided to the databases, communication with other programs running on the same processor, in other crates and on the back-end workstations. Graphical capabilities will be provided by allowing programs to display information on the “back-end” workstation screens through powerful commercial graphical systems.

The back-end computers, the equivalent of today's workstations, provide the development environment, control management, high-level monitoring, and user interfaces. From such standard commercial workstations, programmers can use the software development environment to design and implement the various DAQ software components. These workstations also provide users with the interactive environment to control and monitor the DAQ. The hub of the run-control system, database servers, and ancillary programs such as message loggers and status displays, also run on these machines.

5.3.3 Software development

The complexity of the software system, and the size and the time-scale of the project, indicate that traditional HEP methods of software development would fail to guarantee the required level of reliability, maintainability, and cost-effectiveness. Much remains to be done to find suitable methods for our field of application and much is to be learned in the area of software-engineering tools.

Most aspects of software development clearly span the overall computing environment in ATLAS. Thus a joint effort is needed between the offline and online communities to arrive at common solutions where technically feasible; a detailed discussion of software-engineering and computing-environment issues is presented elsewhere in this document. Computing for triggering and DAQ has, however, a number of specific requirements. Constraints on robustness and reliability are coupled with demanding performance requirements — the system must be capable of operating at high event rates, in parts up to a frequency of ~100kHz. The software development process, including the methodology and related CASE tools, must take into account the following requirements.

- **Support of domain-specific concepts.** Typical examples of concepts specific for T/DAQ computing are the notion of ‘events’ (conditions that can be raised asynchronously), and the
concepts of real time, concurrency (activities which may take place at the same time) and synchronization (concurrent activities may need coordination).

- **Validation.** T/DAQ software is critical and must be validated. Flawed and malfunctioning software may cause data to be lost forever. The validation of the software design, starting from its specification, is of great importance. Formal verification or at least simulation of the specified system is therefore required.

- **Architectural aspects.** Architectural aspects have to be taken into account. For example, processes in the application should be identified automatically, the partitioning of the application in a distributed target environment should be supported. The issue of software integration also has to be considered. Several commercial software products will be used throughout the system. Some limited, specialized applications may need customized software. These should be easily integrated with software produced by a 'standard' development process.

### 5.3.4 Software issues and related R&D

A number of computing and software issues have been discussed in the following domains:

- The need for the software development environment to address also real-time specific issues, such as those outlined in Section 5.3.3.

- The need for a software environment, with emphasis on the operating system and run-time support, which is as common throughout the T/DAQ system as technically possible. Uniformity with the rest of ATLAS computing is also desirable.

- The issue of having offline code running "online". In addition to environmental differences, such as data size and rate, also critical issues of "time to complete" (determining the cost, or the efficiency, of the Event Filter farm) and reliability (errors may not be recoverable "online") have to be addressed.

Several R&D and prototype activities are ongoing in the area of the Trigger and DAQ sub-systems. Some are addressing architectural issues, such as the LVL2 demonstrator programme. In the DAQ area the current prototype programme [5-3] is also addressing, in addition to the more typical DAQ problems, the issues outlined above.

- A software development environment which, while being studied in close contact with the rest of the ATLAS computing, addresses some DAQ specific problems is being studied in the context of the "back-end DAQ" activities.

- The studies and evaluations of operating system environments applicable to DAQ applications are being carried out, with respect to both real-time UNIX systems and the applicability of WindowsNT, in the context of the "front-end DAQ" activities.

- A working group has been set up to specifically address the issues related to the Event Filter.
5.4 References


5-3 ATLAS DAQ "Prototype -1", documented at URL: http://attdoc.cern.ch/Atlas.


A Definitions

Association is a general term to cover connections between classes. Possible uses include inheritance and relationships.

Category a part of a larger model.

Class Diagram shows the classes and their associations. It has something in common with an entity-relationship diagram. It describes the static or structural parts of the system and the relationships between them.

Design Pattern a description of a recurrent problem and of the core of possible solutions.

Domain a part of a larger model which is sufficiently self-contained that it can be handled as a subproject.

Dynamic Model shows the way in which objects change state upon receipt of some event.

Object Interaction Graph (OIG) shows how objects might collaborate to perform a task. It describes the objects and the sequential messages passing between them.

Object Message Diagram (OMD) what an OIG is called in the Unified Notation Language.

Scenario or Use Case is an example of how a system might behave. Books often use scenarios to show only the external behaviour of a system. We treat the term as being an informal (simple text) description of what an OIG is meant to do. It precedes the OIG as the objects may not be clearly identified. An OIG shows how a scenario might be achieved by collaborating objects.
B Acronyms

ADC  Analog to Digital Converter
AOD  Analysis Object Data
ASP  ATLAS Software Process
ATM  Asynchronous Transfer Mode
CASE Computer Aided Software Engineering
DCS  Detector Control System
EM   Electro Magnetic
ESA  European Space Agency
ESD  Event Summary Data
FPGA Field Programmable Gate Array
HEP  High Energy Physics
HPSS High Performance Storage System
ISDN Integrated Services Digital Network
LAN  Local Area Network
LVL1 Level 1 Trigger
LVL2 Level 2 Trigger
LVL3 Level 3 Trigger
MCU  Multiple Connection Unit
MDT  Muon Drift Tube
MSS  Mass Storage System
ODBMS Object Database Management System
ODL  Object Definition Language
OIG  Object Interaction Graph
OMD  Object Message Diagram
OMT  Object Modelling Technique
OO   Object Oriented
ROA  Region Of Activity
ROB  Readout Buffer
ROI  Region of Interest
RPC  Resistive Plate Chamber
SFI  Switch Farm Interface
TGC  Thin Gap Chamber
TRT  Transition Radiation Tracker
UML  Unified Modelling Language
WAN  Wide Area Network
C Members of the ATLAS Collaboration

Armenia
Yerevan Physics Institute, Yerevan
Arapetian A., Grabsky V., Hakopian H., Vartapetian A.

Australia
Research Centre for High Energy Physics, Melbourne University, Melbourne
Fares F., Moorhead G.F., Sevior M.E., Taylor G.N., Tovey S.N.

Australian Nuclear Science and Technology Organisation, Sydney
Alexiev D., Donnelly I.J., Varvell K.E., Williams M.L.

University of Sydney, Sydney
Hashemi-Nezhad R., Peak L., Saavedra A., Ulrichs J.

Austria
Institut für Experimentalphysik der Leopold-Franzens-Universität Innsbruck, Innsbruck
Girtler P., Kienel Ch., Knrenger E., Kuhn D., Rudolph G.

Azerbaijan Republic
Institute of Physics, Azerbaijan Academy of Science, Baku

Republic of Belarus
Institute of Physics of the Academy of Science of Belarus, Minsk
Baturitsky M.A., Bogush A.A., Demchenko A.I., Gavizov A.Z., Gilevsky V.V., Golubev VS Levchuk M.I., Satsunkevich I.S., Shevtsov V.V.

Brazil
Universidade Federal do Rio de Janeiro, COPPE/EE/IF, Rio de Janeiro
Caloba L.P., Galvez-Durand F., Maidantchik C.L., Marroquim F., Seixas J.M., Thome Z.D.

Instituto de Fisica, Universidade de Sao Paulo, Sao Paulo
DaSilva N.C., Dietzsch O., Leite M.A.L., Sakanoue M.H., Takagui E.M., Zandona F.

Canada
University of Alberta, Edmonton

Department of Physics, University of British Columbia, Vancouver
Axen D., Mayer J.K., Orr R.S.

University of Carleton/C.R.P.P., Carleton
Armitage J., Dixit M., Dubéau J., Estabrooks P., Losty M., Neuheimer E., O'Neil M., Oakham G.

Group of Particle Physics, University of Montreal, Montreal

Department of Physics, University of Toronto, Toronto
Bailey D.C., Bhadra S., Martin J.F., Sinervo P.K., Stairs G.G., Trischuk W.

TRIUMF, Vancouver
Astbury A., Birney P., Hodges T., Langstaff R., Oram C., Roberts B., Rosvick M., Wellisch H.P.

University of Victoria, Victoria
CERN
European Laboratory for Particle Physics (CERN), Geneva

Czech Republic
Academy of Sciences of the Czech Republic, Institute of Physics, Prague

Charles University, Faculty of Mathematics and Physics, Prague

Czech Technical University in Prague, Faculty of Nuclear Sciences and Nuclear Engineering, Faculty of Mechanical Engineering, Prague
Jakubek J., Kubasta J., Macha L., Ota J., Pospisil S., Sinor M., Sopko B., Tomiak Z.

Denmark
Niels Bohr Institute, University of Copenhagen, Copenhagen
Dam M., Hansen J.D., Hansen J.R., Hansen P., Rensch B.

Finland
Helsinki Institute of Physics, Helsinki
Jalas P., Schulman T.

France
Laboratoire d'Annecy-le-Vieux des Particules (LAPP), IN2P3-CNRS, Annecy-le-Vieux

Université Blaise Pascal, IN2P3-CNRS, Clermont-Ferrand

Institut des Sciences Nucléaires de Grenoble, IN2P3-CNRS-Université Joseph Fourier, Grenoble

Centre de Physique des Particules de Marseille, IN2P3-CNRS, Marseille
Laboratoire de l’Accélérateur Linéaire, IN2P3-CNRS, Orsay

LPNHE, Universités de Paris VI et VII, IN2P3-CNRS, Paris

CEA, DSM/DAPNIA, Centre d’Etudes de Saclay, Gif-sur-Yvette

Republic of Georgia
Institute of Physics of the Georgian Academy of Sciences, Tbilisi

Tbilisi State University, Tbilisi
Chiladze B., Djobava T., Khelashvili A., Khubua J., Liparteliani A., Metreveli Z., Mosidze M., Salukvadze R.

Germany
Physikalisches Institut, Universität Bonn, Bonn
Desch K., Fischer P., Geich-Gimbel C., Hilger E., Meuser S., Ockenfels W., Raith B., Wernecke N.

Institut für Physik, Universität Dortmund, Dortmund
Becker C., Fuss J., Goessling C., Lisowski B., Luthaus P., Wunstorf R.

Fakultät für Physik, Albert-Ludwigs-Universität, Freiburg

Institut für Hochenergiephysik der Universität Heidelberg, Heidelberg
Geweniger C., Hanke P., Kluge E.-E., Mass A., Putzer A., Titel K., Wunsch M.

Institut für Informatik, Friedrich-Schiller-Universität Jena, Jena
Doering V., Erhardt W., Kammel P., Reinsch A.

Institut für Physik, Johannes-Gutenberg-Universität Mainz, Mainz

Lehrstuhl für Informatik V, Universität Mannheim, Mannheim

Sektion Physik, Ludwig-Maximilian-Universität München, München

Max-Planck-Institut für Physik, München
Fachbereich Physik, Universität Siegen, Siegen
Gillessen G., Holder M., Kreutz A.
Fachbereich Physik, Bergische Universität, Wuppertal

Greece
Athens National Technical University, Athens
Dris M., Filippas A., Fokitis E., Gazis E., Katsoufis E., Maltezos S., Papadopoulos T.

Athens University, Athens
Giokaris N., Ioannou P., Kourkoumelis C., Papadopoulos I., Tatis S., Tzanakos G.S.

Aristotle University of Thessaloniki, Thessaloniki
Bouzakis C., Chardalas M., Chouridou S., Dedousis S., Gavras G., Lagouri Th., Liolios A., Paschalias P., Petridou C., Sampsonidis D., Sergiadis G., Zamani M.

Israel
Department of Physics, Technion, Haifa
Dado S., Goldberg J., Lupu N.

Raymond and Beverly Sackler Faculty of Exact Sciences, School of Physics and Astronomy, Tel-Aviv University, Tel-Aviv

Department of Particle Physics, The Weizmann Institute of Science, Rehovot

Italy
Dipartimento di Fisica dell' Università della Calabria e I.N.F.N., Cosenza

Laboratori Nazionali di Frascati dell' I.N.F.N., Frascati
Bilokon H., Chiacrella V., Curatolo M., Esposito B., Ferrer M., Maccarrone G., Pace E., Pepe-Artarelli M., Spitalieri M., Zuffranieri F.

Dipartimento di Fisica dell' Università di Genova e I.N.F.N., Genova

Dipartimento di Fisica dell' Università di Lecce e I.N.F.N., Lecce
Cesaroni F., Creti P., Gorini E., Granaganolo F., Palmara O., Panareo M., Petrella S., Primavera M.

Dipartimento di Fisica dell' Università di Milano e I.N.F.N., Milano

Dipartimento di Scienze Fisiche, Università di Napoli 'Federico II' e I.N.F.N., Napoli
Allosio A., Alviggi M.G., Cevenini F., Chiefari G., De Asmundis R., Drago E., Merola L., Napolitano M., Patricelli S.

Dipartimento di Fisica Nucleare e Teorica dell' Università di Pavia e I.N.F.N., Pavia

Dipartimento di Fisica dell' Università di Pisa e I.N.F.N., Pisa

Dipartimento di Fisica dell' Università di Roma 'La Sapienza' e I.N.F.N., Roma
Dipartimento di Fisica dell' Università di Roma 'Tor Vergata' e I.N.F.N., Roma
Camarri P., Cardarelli R., Di Ciaccio A., Santonico R.
Dipartimento di Fisica dell' Università di Roma 'Roma Tre' e I.N.F.N., Roma
Bacci C., Ceradini F., Orestano D., Pastore F.
Dipartimento di Fisica dell' Università di Udine, Gruppo collegato di Udine I.N.F.N. Trieste, Udine
Cauz D., D'Auria S., De Angelis A., Paulettta G., Santi L., Scuri B., Waldner F., del Papa C.

Japan
Department of Information Science, Fukui University, Fukui
Kawaguti M., Tanaka S.

Hiroshima Institute of Technology, Hiroshima
Asai M.

Department of Physics, Hiroshima University, Higashi-Hiroshima
Iwata Y., Ohsugi T.

KEK, National Laboratory for High Energy Physics, Tsukuba

Department of Physics, Faculty of Science, Kobe University, Kobe
Kawagoe K., Nozaki M., Takeda H.

Kyoto University of Education, Kyoto-shi
Takashima R.

Naruto University of Education, Naruto-shi
Nagamatsu M., Yoshida H.

Department of Physics, Faculty of Science, Shinshu University, Matsumoto
Takeshita T.

International Center for Elementary Particle Physics, University of Tokyo, Tokyo
Hasegawa Y., Imori M., Kawamoto T., Kobayashi T., Mashimo T.

Physics Department, Tokyo Metropolitan University, Tokyo
Fukunaga C., Hamatsu C.

Tokyo University of Agriculture and Technology, Department of Applied Physics, Tokyo
Emura T.

Kazakhstan
High-Energy Physics Institute of the Kazakh Academy of Sciences, Almaty
Boos E., Pokrovsky N., Zhautykov B.

Netherlands
FOM - Institute SAF NIKHEF and University of Amsterdam/NIKHEF, Amsterdam

University of Nijmegen/NIKHEF, Nijmegen

Morocco
Faculté des Sciences Ain Chock, Université Hassan II, Casablanca, and Université Mohamed V, Rabat
Chakir H., Cherkaoui R., Hoummad A., Saidi H., Sayouty E.

Norway
University of Bergen, Bergen
Eigen G., Frodesen A.G., Klovning A., Stug B.
University of Oslo, Oslo
Bugge L., Buran T., Kristiansen H., Nygaard E., Stapnes S.

Poland
Henrik Niewodniczanski Institute of Nuclear Physics, Cracow
Gadomski S., Godlewski J., Hajduk Z., Kisielewski B., Korcyl K., Malecki P., Moszczynski A.,
Olszowska J., Richter-Was E., Sobala A.

Faculty of Physics and Nuclear Techniques of the Academy of Mining and Metallurgy, Cracow

Portugal
Laboratorio de Instrumentação e Física Experimental de Partículas (University of Lisboa, University of Coimbra,
University Católica-Figueira de Foz and University Nova de Lisboa), Lisbon
Amaral P., Amorim A., Carvalho J., Casarejos E., David M., Gomes A., Gomes J., Ivanychkiv I.,

Romania
Institute of Atomic Physics, Bucharest
Arseneuc R., Boldea V., Caprini I., Caprini M., Constantin F., Constantinescu S., Dita P., Dita S.,

Russia
Institute for Theoretical and Experimental Physics (ITEP), Moscow
Artamonov A., Epchtein V., Gorbunov P., Gurin R., Jemanov V., Khovansky V., Koutchenkov A.,
Kruchinin S., Maslennikov A., Ryabinin M., Shatalov P., Tsoukerman I., Zaitsev V., Zeldovich S.

P.N. Lebedev Institute of Physics, Moscow
Akinov A., Baranov S., Belov M., Blagov M., Gavrilenko I., Komar A., Konovalov S., Kopytine M.,
Mouraviev S., Popov L., Shikyan A., Shmelev A., Snisarev A., Speransky M., Sulin V., Tikhomirov V.,
Vasilieva L., Yakimenko M.

Moscow Engineering and Physics Institute (MEPhI), Moscow
Bondarenko V., Dolgoshein B., Konstantinov A., Nevski P., Romaniouk A., Semenov S., Smirnov S.,
Sosnovtzev V.

Moscow State University, Institute of Nuclear Physics, Moscow
Bashindjagian G.L., Basiladze S.G., Chudakov E.A., Erasov A.B., Grishkevich Y., Karmanov D.E.,
Kramarenko V.A., Larichev A.N., Melikhov D.I., Merkin M.M., Nikitin N.V., Rizatdinova F.K.,
Selikov A.V., Sivoklokov S.Yu., Smirnova L.N., Zhukov V.Yu., Zverev E.G.

Budker Institute of Nuclear Physics (BINP), Novosibirsk
Batrakov A., Chilingarov A., Fedotov M., Gaponenko I., Klimenko S., Kollegov M., Kozlov V., Kuper E.,
Merzylak Y., Panin V., Sharov A., Telnov V., Tikhonov Y., Velikhzhanin Y.

Institute for High Energy Physics (IHEP), Protvino
Amelin D.V., Ammosov V.V., Antipov Yu.M., Batarin V., Bogoliubsky M.Yu., Borissov A.A., Borissov E.,
Bozko N.I., Bryzgalov V.V., Chekulaev S.V., Denisov S.P., Dushkin A.Yu., Fakhrotdinov R., Fenyuk A.B.,
Kirsanov M.M., Klyukhin V., Kojine A., Kononov A.I., Konstantinov V., Kopikov S.V., Korotkov V.A.,
Kostrikov M.E., Kostyukhin V.V., Kravtsov V.I., Kulemzin A., Kurchaninov L.L., Lapin VV.,
Levitsky M.L., Los S., Maximov V., Miagkov A.G., Mikhailin V.N., Minaenko A.A., Moiseev A.M.,
Onuchin V.A., Plesakach A.V., Salomatkin Yu.I., Senko V.A., Shein I., Soldatov A.P., Solodkov A.A.,
Solovianov O.V., Starchenko E.A., Sviridov Yu., Sytnik V.V., Tchmil V., Tchoutonov A., Tikhonov VV.,

Petersburg Nuclear Physics Institute (PNPI), Gatchina, St. Petersburg
Fedin O., Filimonov V., Gavrilo V., Ivochkin V., Khomoutnikov V., Kolos S., Krivchitch A., Lohak I.,
Malev V., Nadtochy A., Patrigan A., Prokofiev D., Riabov J., Schegelkev S., Soloviev I., Spiridenkov E.,
Zalite A.
JINR
Joint Institute for Nuclear Research, Dubna

Slovak Republic
Bratislava University, Bratislava, and Institute of Experimental Physics of the Slovak Academy of Sciences, Kosice

Slovenia
Joseph Stefan Institute and Department of Physics, University of Ljubljana, Ljubljana
Cindro V., Filipcic A., Kramberger G., Mikuz M., Tadel M., Zontar D.

Spain
Institut de Fisica d'Altes Energies (IFAE), Universitat Autonoma de Barcelona, Bellaterra, Barcelona

Physics Department, Universidad Autonoma de Madrid, Madrid
Barreiro F., Del Peso J., Hervas L., Labarga L.

Instituto de Fisica Corpuscular (IFIC), Centro Mixto Universidad de Valencia-CSIC, Burjasot, and Departamento de Informatica y Electronica, Universidad de Valencia, Burjasot, Valencia

Sweden
Fysika institutionen, Lunds universitet, Lund

Royal Institute of Technology (KTH), Stockholm
Carlson P., Clement C., Lund-Jensen B., Pearce M., Soderqvist J., Vanyashin A.

University of Stockholm, Stockholm

Uppsala University, Department of Radiation Sciences, Uppsala
Bingefors N., Botner O., Brenner R., Bystrom O., Ekelof T., Gustafsson L., Hallgren A., Kullander S., Staaf P.

Switzerland
Laboratory for High Energy Physics, University of Bern, Bern

Section de Physique, Universite de Genève, Geneva
Turkey
Department of Physics, Bogaziçi University, Istanbul

United Kingdom
School of Physics and Space Research, The University of Birmingham, Birmingham

Cavendish Laboratory, Cambridge University, Cambridge

Department of Physics and Astronomy, University of Edinburgh, Edinburgh
Boyle O., Candlin D.J., Candlin E.R.S., Knowles L.G., Playfer S.M.

Department of Physics and Astronomy, University of Glasgow, Glasgow

School of Physics and Chemistry, Lancaster University, Lancaster
Brodbbeck T.J., Henderson R.C.W., Hughes G.H., Komorowski T., Ratoff P.N., Sloan T.

Department of Physics, Oliver Lodge Laboratory, University of Liverpool, Liverpool

Department of Physics, Queen Mary and Westfield College, University of London, London

Department of Physics, Royal Holloway and Bedford New College, University of London, Egham
Blair G.A., George S., Green B.J., Medcalf T., Strong J.A.

Department of Physics and Astronomy, University College London, London
Bignall P., Clarke P., Cranfield R., Crane G., Esten M., Jones T., Lane J., Sherwood P.

Department of Physics and Astronomy, University of Manchester, Manchester

Department of Physics, Oxford University, Oxford

Rutherford Appleton Laboratory, Chilton, Didcot

Department of Physics, University of Sheffield, Sheffield
Booth C.N., Buttar C.M., Cartwright S.L., Combley F.H., Lehto M.H., Sellin P.J., Spooner N.J.C., Thompson L.F.

United States of America
State University of New York at Albany, New York
Alam S., Atahir B., Mahmood A., Ovunc S., Severini H., Timm S., Wappler F., Zhichao L.

Argonne National Laboratory, Argonne, Illinois
University of Arizona, Tucson, Arizona
Cheu E., Johns K., Loch P., Rutherford J., Savine A., Shaver L., Shupe M., Steinberg J., Tompkins D.

Department of Physics, The University of Texas at Arlington, Arlington, Texas
De K., Gallas E., Li J., Sawyer L., Stephens R., White A.

Lawrence Berkeley Laboratory and University of California, Berkeley, California

Department of Physics, Boston University, Boston, Massachusetts
Hazen E., Shank J., Simmons E., Whitaker J.S., Zhou B.

Brandeis University, Department of Physics, Waltham, Massachusetts
Behrends S., Bensinger J.R., Blocker C., Cunningham J., Kirsh L.E., Lamoureux J., Wellenstein H.

Brookhaven National Laboratory (BNL), Upton, New York

University of Chicago, Enrico Fermi Institute, Chicago, Illinois
Anderson K., Blucher E., Evans H., Glensk Iny D., Merritt F., Oreglia M., Pilcher J., Pod E., Sanders H., Shochet M., Turcot A.

Nevis Laboratory, Columbia University, Irvington, New York
Cartiglia N., Cunitz H., Dodd J., Gara J., Leltchouk M., Parsons J., Seman M., Shaevitz M., Sippach W., Willis W.

Department of Physics, Duke University, Durham, North Carolina

Department of Physics, Hampton University, Virginia
Baker K.

Department of Physics, Harvard University, Cambridge, Massachusetts
Feldman G.J., Franklin M.E.B., Huth J., Oliver J.

Indiana University, Bloomington, Indiana
Hanson G., Li Z., Leeering F., Ogren H., Rust D.R.

University of California, Irvine, California
Lankford A.J., Heckett C., Pier S., Schernau M., Stoker D.

Massachusetts Institute of Technology, Department of Physics, Cambridge, Massachusetts

Michigan State University, Department of Physics and Astronomy, East Lansing, Michigan

University of New Mexico, New Mexico Center for Particle Physics, Albuquerque
Gold M., Goff G., Matthews J., Seidel S.

Physics Department, Norfolk State University, Virginia
Khandaker M., McFarlane K., Punjabi V., Salgado C.W.

Physics Department, Northern Illinois University, DeKalb, Illinois
Fortin M., Sirotenko V.I., Willis S.E.

Department of Physics and Astronomy, University of Oklahoma
Gutierrez P., McMahon T., Nemati B., Skubic P., Snow J., Strauss M.

Department of Physics, University of Pennsylvania, Philadelphia, Pennsylvania
Dressandt N., Keener P., Newcomer F.M., Van Berg R., Williams H.H.
University of Pittsburgh, Pittsburgh, Pennsylvania
Cleland W.E., Cleren M.

Department of Physics and Astronomy, University of Rochester, Rochester, New York

Institute for Particle Physics, University of California, Santa Cruz, California

Department of Physics, Southern Methodist University, Dallas, Texas
Coan T.E., Olness F., Skwarnicki T., Stroynowski R., Teplitz V.

Tufts University, Medford, Massachusetts
Mann A., Millburn R., Napier A., Sliwa K.

High Energy Physics, University of Illinois, Urbana, Illinois
Downing R.W., Errede D., Errede S., Haney M.J., Simaitis V.J., Thaler J.

Department of Physics, Department of Mechanical Engineering, University of Washington, Seattle, Washington
Burnett T.H., Chaloupka V., Cook V., Daly C., Davisson R., Forbush D., Guldenmann H., Lubatti H.J.,
Mockett P.M., Reinhall P., Rothberg J., Wasserbaech S., Zhao T.

Department of Physics, University of Wisconsin, Madison, Wisconsin
Fasching D., Gonzalez S., Jared R.C., Pan Y.B., Scott I.J., Walsh J., Wu S.L., Yamartino J.M., Zobernig G.

This document has been prepared with Release 5 of the Adobe FrameMaker® Technical Publishing System using the Technical Design Report template prepared by Mario Ruggieri of the Information and Programming Techniques Group, ECP Division, CERN, according to requirements from the ATLAS collaboration.

To facilitate multiple author editing and electronic distribution of documents, only widely available fonts have been used. The principal ones are:

Running text: Palatino 10.5 point
Chapter headings: Helvetica Bold 18 point
2nd, 3rd and 4th level headings: Helvetica Bold 13, 11.5 and 10.5 point respectively
Figure and table captions: Helvetica 9 point