

ATLAS Data Challenge 1

(Only Phase 1 so far)

The ATLAS DC1 Task Force¹

Abstract:

A major feature of the current computing activities (DC1) in ATLAS is the preparation and deployment of the software required for the production of large event samples for the High Level Trigger (HLT) and physics communities, and the actual production of those samples. It should be noted that it is not an option to “run everything at CERN” even if we wanted to; the resources are not available at CERN to carry out the production on a reasonable time-scale. We have therefore had to face the great challenge of organising and then carrying out this large-scale production at a significant number of sites around the world. However, the benefits of this are manifold: apart from realising the required computing resources, this exercise builds worldwide momentum for ATLAS computing as a whole.

Much has been learned from this first phase of DC1, and much will doubtless be learned over the next few months. However, we can already be rather confident that ATLAS will be able to marshal worldwide resources in an effective way; let us hope that the Grid will make it all rather easy.

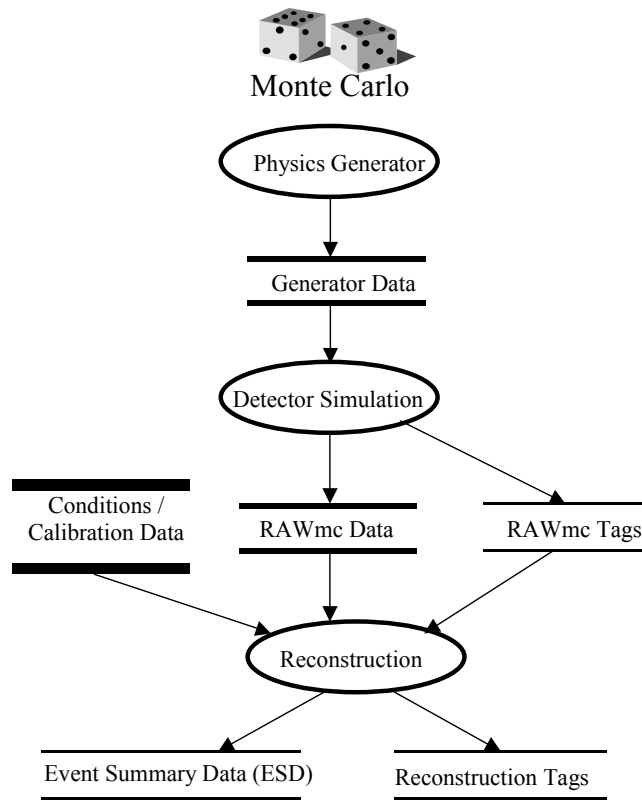
1.) Introduction

During the Phase-1 of the LCG project (2001-2005) a series of Data Challenges will be performed. The experience gained during these exercises will be used to formulate the ATLAS Computing TDR, which is due, according to the present planning, at the end of 2003. The Grid technologies promise several advantages for a multinational, geographically distributed project: they allow for a uniform infrastructure of the project computing-wise, simplify the management and coordination of the resources while potentially decentralizing such tasks as software development and analysis, and last, but not least, the Grid is an affordable way to increase the computing power. If the ATLAS Data Challenges will demonstrate that usage of the Grid, indeed, gives all those advantages, the collaboration should become committed to “gridification” of its sites and tools, by making use of the best available Grid middleware.

During the LHC preparation phase, all experiments have large needs for simulated data, to design and optimise the detectors. This “Monte Carlo” simulation is done in the following steps:

- Particles emerging from the collisions (called collision final state or simply final state) are generated using programs usually based on physics theories and phenomenology (called generators);
- The particles of the generated final state are transported through the virtual detector according to the known physics laws governing the passage of particles through matter;
- The resulting interactions with the sensitive elements of the detector are converted into rates of electronic counters (digitisation) similar to those produced by the real detector;
- The events are reconstructed.
- The (Monte Carlo) generated information (sometimes called *truth*) is saved for comparison with the reconstructed information.

¹ Please send your comments to : Alois.Putzer@cern.ch



2.) The LCG Project

The job of the LHC Computing Grid Project – LCG – is to prepare the computing infrastructure for the simulation, processing and analysis of LHC data for all four of the LHC collaborations. This includes both the common infrastructure of libraries, tools and frameworks required to support the physics application software, and the development and deployment of the computing services needed to store and process the data, providing batch and interactive facilities for the worldwide community of physicists involved in LHC.

The first phase of the project, from 2002 through 2005, is concerned with the development of the application support environment and of common application elements, the development and prototyping of the computing services and the operation of a series of computing data challenges of increasing size and complexity to demonstrate the effectiveness of the software and computing models selected by the experiments.

This first phase will conclude with the production of a Computing System Technical Design Report, providing a blueprint for the computing services that will be required when the LHC accelerator begins production. This will include capacity and performance requirements, technical guidelines, costing models, and a construction schedule taking account of the anticipated luminosity and efficiency profile of the accelerator.

A second phase of the project is envisaged, from 2006 through 2008, to oversee the construction and operation of the initial LHC computing system.

3.) ATLAS Data Challenges

For all data challenges it is essential to have physics content in order to bring the physicists community into the exercise and for a more sensible validation of the software. It is important to mention that for DC1, in 2002, a major goal is to provide simulated data to the High Level Trigger (HLT) community that has to prepare its own Technical Design Report (TDR) by mid 2003.

The goals of the ATLAS Data Challenges are the validation of the Computing Model, of the complete software suite, of the data model, and to ensure the correctness of the technical choices to be made. It is understood that these Data Challenges should be of increasing complexity and will use the software which will be developed in

the LCG project, to which ATLAS is committed, as well as the Grid middleware being developed in the context of several Grid projects like EU Data Grid or GridPP. The results of these data challenges will be used as input for a Computing Technical Design Report and for preparing a Memorandum of Understanding in due time.

Data Challenge 1 runs from April 2002 to early part of 2003. It is divided into three phases:

- In the first phase, April-August 2002, we have put in place the infrastructure and the production tools to be able to run the production worldwide; a major production was run (see below).
- In the second phase, which will start in October 2002, we intend to produce the pile-up data.
- The reconstruction and the analysis of the data will be performed in the first half of 2003.

The event generation can use several event generators (e.g. Pythia, Herwig, Isajet, etc.) and can run either in the Fortran ATLSIM framework or in the ATHENA framework. The fast simulation, ATLFAST, has been moved to OO. It runs in the ATHENA framework.

The current detector simulation code called DICE is Fortran based, uses GEANT 3.21 to track the events through the detector and runs in the ATLSIM framework. Events are written out in the form of ZEBRA banks. Most of the reconstruction programs have now moved to OO, even if some packages are still in Fortran. The new reconstruction uses the ATHENA framework and in the current situation input data can come from ZEBRA, ROOT-IO or Objectivity.

Essential components required for ATLAS Monte Carlo production are the associated bookkeeping and meta-data services.

4.) Resources used in DC1 Phase 1

In DC1 phase1 the data needed for the High Level Trigger (HLT) TDR were generated (i.e. 4-vector production using PYTHIA), followed by full simulation, after some selection, of the ATLAS detector response using ATLSIM (Dice, GEANT3). Due to the huge amount of computing time needed it was essential to make use of the computing resources available in different ATLAS institutes.

4.1 Countries participating

The following 39 institutes in 18 countries participated in DC1 phase 1:

Australia (Melbourne)

Austria (Innsbruck)

Canada (Alberta, CERN)

CERN

Czech Republic (Prague)

France (Grenoble + Marseille; using Lyon)

Germany (München; using FZK)

Israel (Weizmann)

Italy (CNAF Bologna, Frascati, Milano, Napoli, Roma)

Japan (Tokyo)

NorduGrid : Denmark, Norway, Sweden (Bergen, Grendel, Ingvar, ISV, LSCF, Lund, NBI, Oslo)

Russia (Dubna, ITEP Moscow, MSU Moscow, Protvino)

Spain (Valencia)

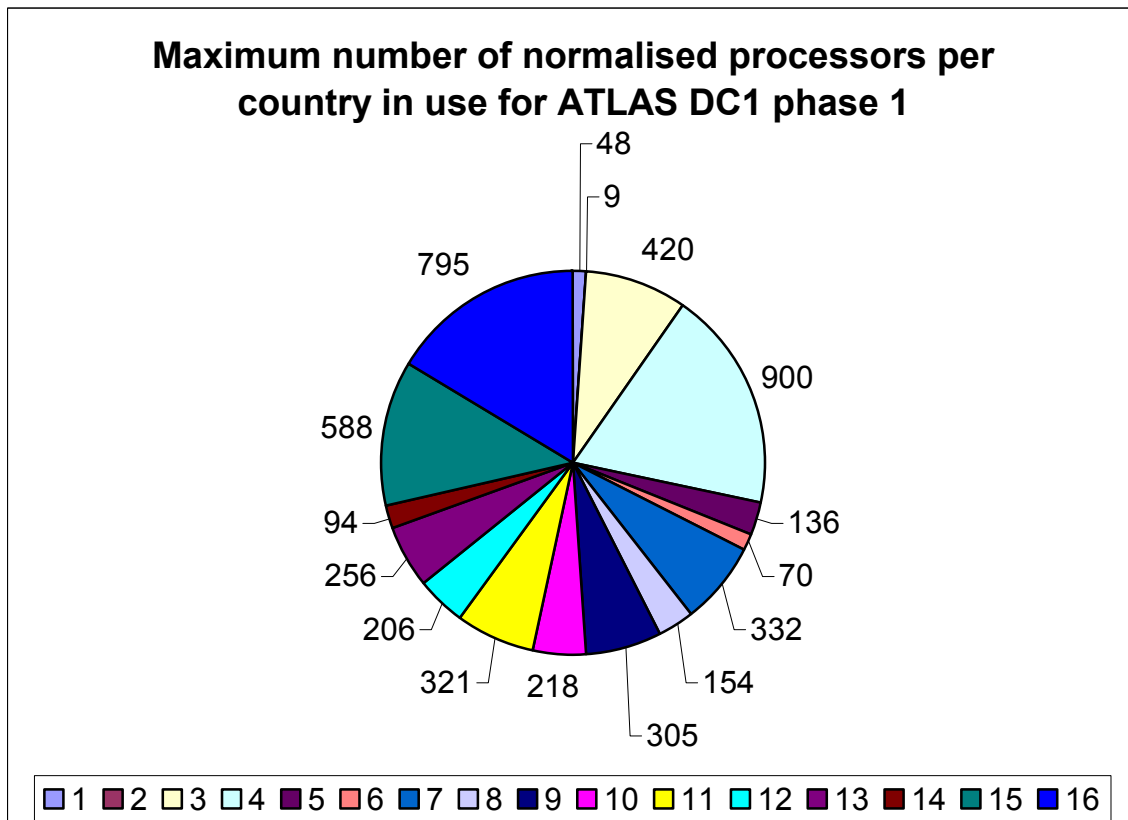
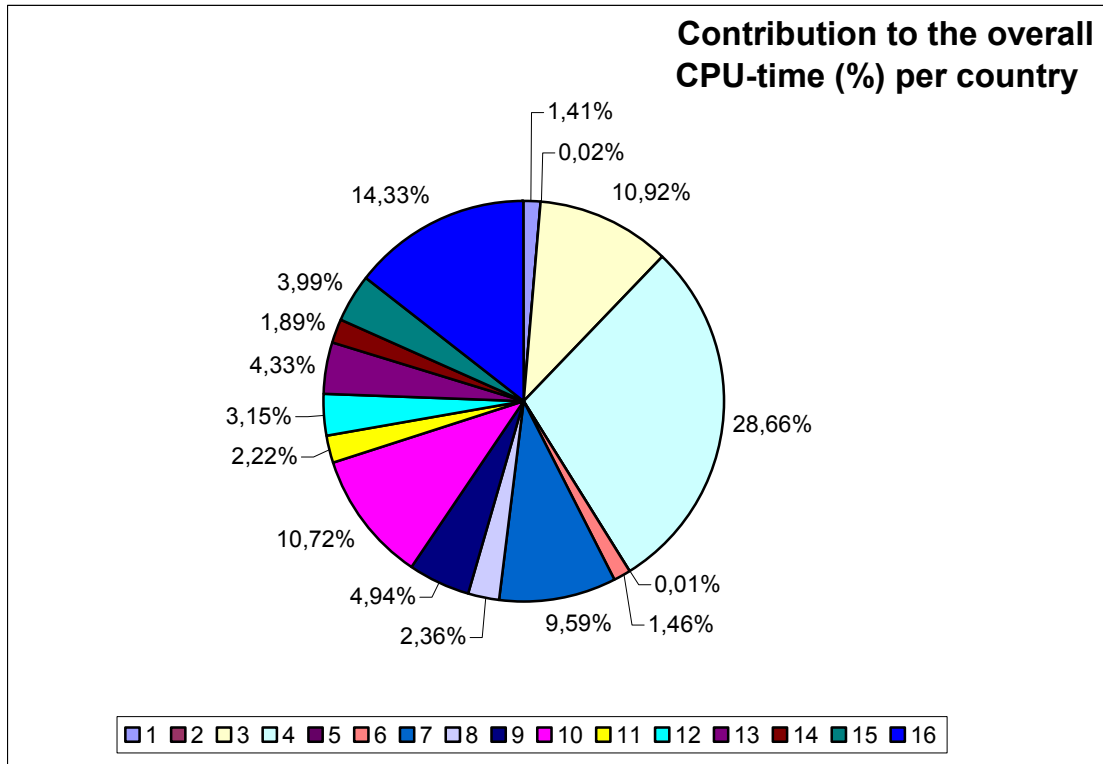
Taiwan (Taipei)

UK (Cambridge, Glasgow, Lancaster, Liverpool, RAL)

USA (Arlington, BNL, LBNL, Oklahoma)

More countries and sites will join later this year.

4.2 Resources available for DC1 phase 1



In what follows we use the following conversion factor:

1 Normalised Computing Unit (NCU) corresponds to 1 Pentium III 500 MHz equivalent to 21 SpecInt95 (SI95).

The numbers of processors per site varied between 9 and 900. At peak time we used worldwide ~3200 processors (~5000 NCUs) in 39 institutes located in 18 countries. This corresponds to ~110 kSI95 or 50% of the CPU power estimated for one Regional Facility at the LHC start-up (2007). The hardware investment made by those institutes in the last 12 months corresponds roughly to 50 % of the yearly hardware investment needed from 2006 onwards for the non-CERN part of the ATLAS Offline Computing.

4.3 Data Samples

Event type	Output size/event (MB)	CPU-time/event (SI95sec)
Single Particle	0.05	300
Minbias	1.00	4000
Di-jet event	2.40	13000

Average numbers for the different samples

During Phase 1 of DC1, about 50 million events in total were generated via PYTHIA; about 50 million events in total were passed through detailed detector simulation via ATLSIM. About 40 million of the latter were single-particle events (muons, photons, electrons, pions), the remaining about 11 million ones complete physics events. This exercise took 74000 CPU-days and produced a total data volume of about 32 Tbytes in about 35000 partitions.

The production requests from the High-Level-Trigger community were originally collected by Monika Wielers² and organised into three main parts: validation samples (very high priority), high-statistics samples (mostly high priority), and medium-statistics samples (ranging from low to high priority). In the course of DC1/1, the information contained in that page was re-arranged and a new working page³ set up to monitor the progress of the production activities. In addition to the original information (physics contents, simulation specifications, number of events, priority), this page contains also organisational (dataset numbers, groups-in-charge, status, etc.) and statistical (numbers of generated/simulated events, time to process one/all events, etc.) information relevant for the individual sub-samples. Most of this information will eventually be accessible from the bookkeeping database (AMI; see 6.5.1).

The data at CERN are stored in the CASTOR system. Generated events are located in directories
`/castor/cern.ch/atlas/project/dc1/evgen/data/<dataset_number>`

simulated ones in
`/castor/cern.ch/atlas/project/dc1/simul/data/<dataset_number>`

The samples assigned the highest priority were the validation samples⁴. They consist of single-particle events, jet-scan samples, and some physics event channels taken from old TDR tapes. About 740k events were processed and 110 GBytes of data were produced, the time needed being about 900 CPU days.

The most challenging part, w.r.t. CPU and data storage requirements, was the production of the high-statistics samples⁵. They consist of 36 million single-muon events, about 5 million di-jet events of different E_T (hard scattering) cuts (applying particle-level filtering or not), and 1 million minimum-bias events simulated with different $|\eta|$ cuts. The data volume of the whole sample amounts to about 15 TBytes, the total CPU time needed to about 44000 days. Note that not all the produced data will be stored in the CERN CASTOR system; about 10 TBytes will be kept at different distributed production sites.

² http://mwielers.home.cern.ch/mwielers/productions_new.html

³ http://atlasinfo.cern.ch/ATLAS/GROUPS/SOFTWARE/DC/DC1/DC1_1/production_requests.html

⁴ http://atlasinfo.cern.ch/ATLAS/GROUPS/SOFTWARE/DC/DC1/DC1_1/validation/validation_samples.html

⁵

http://atlasinfo.cern.ch/ATLAS/GROUPS/SOFTWARE/DC/DC1/DC1_1/highStat/high_statistics_samples.html

The medium-high statistics samples comprise production requests by various subgroups of the HLT community: the e/γ , Level-1, jet/ETmiss, B-physics, b-jet, and muon trigger groups; and a sample of about 80k single pions. The information is accessible from the web⁶. The e/γ samples contain a huge production of single-electron (1.1 million) and single-photon (1.6 million) events at different energies and η values. Sub-samples of the B-physics trigger and b-jet trigger samples were simulated for the Inner Detector only, the rest either with the "central" detector (ID+Calorimeters; e.g., the e/γ single-particle production) or the full detector. All the generated events for the B-physics trigger group were taken from existing TDR tapes. All the about 7 million simulated events correspond to a data volume of about 9 TBytes, the total CPU time necessary to process them was 30000 days.

In summary, the total estimated data volume produced during DC1/1 is about 24 Tbytes and about 8 TBytes for generated events; the total CPU time necessary to generate all the events was about 1000 days, the time to simulate all the events about 74000 days.

5.) Software Distribution

The ATLAS software source code is maintained at CERN in a CVS repository and then installed and compiled in a public AFS directory, under the ATLAS tree. The compilation process is done on Linux machines running CERN RedHat Linux 6.1. Users with a good network connection and access to AFS may use executables and the data files directly linking them from CERN. This approach is anyway not indicated for remote sites with a bad connection to CERN or without access to AFS. For this purpose, starting from the release 3.0.0, a set of RPM packages has been produced, in order to install the full ATLAS software distribution on machines both with and without AFS.

The RPM kit has been designed to be used on standard as well as on EDG machines, in order to fulfil the requirements of the Data Challenge 1. During the second phase of the DC1, in fact, the reconstruction is planned to be done by using the EDG tools. Each ATLAS software release is packaged into RPM format. The kit, along with the installation script, is downloadable⁷ via secure web connection or, otherwise, from the EDG site⁸.

The general criteria, followed during the package architecture development phase, have been to build a self-consistent, LINUX release independent distribution. To fulfil these requirements the RPMs have been designed to keep the same directory structure as in the CERN repository and to include the reference gcc compiler (gcc v2.95.2), the ROOT version used for the build of the release and the required libraries not part of the ATLAS software. To be consistent with the reference software, produced at CERN, the executables and libraries included in the kit are the exact copies of the files stored in the public AFS software repository.

The packages are organized in a set of base tools, required for all the installations, and several additional components. A minimal installation should provide at least the following items:

- the set-up and management scripts;
- the official ATLAS compilers;
- the ROOT version using during the compilation phase;
- the required libraries not part of the ATLAS software (external packages).

This corresponds to the ATLAS-conf, ATLAS-tools, ATLAS-release, ATLAS-compilers, ATLAS-root and ATLAS-external RPMs. If the system compiler is gcc v2.95.2 users may choose not to install the ATLAS-compiler package. Other packages are anyway required to generate, simulate and reconstruct the data; therefore it is highly recommended that the full set of RPMs be installed on each machine.

The kit installs itself under the directory /opt/ATLAS, using about 1 GB of disk space. Relocation is also possible, providing that the change of the root directory of the kit, from /opt/ATLAS to some other place, is also reflected in the configuration scripts, by editing them after the installation. For convenience, a relocation script is included in the kit, under the /opt/atlas/etc directory. To work with this kit, at first users must configure the environment via the set-up script (/opt/atlas/etc/atlas.shrc). After this is done, the applications are ready to be executed correctly. Some examples on how to run a simulation job are included in the kit in the ATLAS-DC1 package.

⁶http://atlasinfo.cern.ch/ATLAS/GROUPS/SOFTWARE/DC/DC1/DC1_1/mediumStat/medium_statistics_samples.html

⁷ <https://classis01.roma1.infn.it/atlas-farm/atlas-kit>

⁸ <http://datagrid.in2p3.fr/distribution/applications/wp8/atlas>

The current RPM suite (ATLAS software v3.2.1) has proven to be robust and efficient during the first part of the Data Challenge 1, in which several millions of events have been successfully simulated on different machines and LINUX distributions.

Most of the countries and sites have installed the software using the official set of RPMs, but other types of installations have also been used in some sites for the DC1 production. In particular a first one based on a full mirroring of the distributions, directly from the CERN AFS repository, and a second one from a different set of RPMs, developed by the Nordic Countries and used within the NorduGrid test-bed.

6.) Generation and Simulation

6.1 Event Generation

The generation of all event samples was done at CERN using Pythia 6.203⁹ running inside ATHENA (ATLAS releases 3.2.0 and 3.2.1). The events were converted into HepMC¹⁰ and then written out into ROOT I/O using the ATHENA Root conversion service.

The default Pythia parameters were used with the following exceptions:

The multiple interaction model is used for underlying and minimum bias events MSTP (82)=4 and PARP (82)=2.2, since this gives better agreement with the CDF data¹¹. In addition the fragmentation parameters were set to: MSTJ (11)=3, PARJ (54)=-0.07 and PARJ (55)=-0.006. MSTJ (22)=2 was used to ensure stable K^0 and Λ .

The following samples were generated:

A "**jet**" sample with the following processes activated; 1,2,11,12,13,28,53,68,81,82,14 and 29. No kinematics' cuts were applied, except for a minimum transverse momentum CKIN (3), whose value can be found in the full list of events. The largest sample was generated with CKIN (3) =17. This sample is dominated by the $2 \rightarrow 2$ QCD processes such as $gg \rightarrow gg$.

A "**minimum bias**" sample with MSEL=1. These events are used for pile-up for which it is recommended that the cross section should be set to 67 mb¹². To generate minimum bias events for DC1 the PYTHIA generator version 6.203 is used. The parameters used for the event generation represent the best tuning for energies up to the Tevatron energies.

Single W sample: Inclusive W production using process 2. For the sample, where W is forced to decay to $\tau\nu$, the kinematics' range is restricted by setting CKIN (1)=71 and CKIN (2)=91. For the sample, where W is forced to decay to $e\nu$, only CKIN (1)=71 is used.

Single Z samples: Three sets forcing the Z to decay to e^+e^- , $\mu^+\mu^-$ and $\tau^+\tau^-$ were generated. Process 1 was activated with MSTP (43)=2 so that only the Z boson contributes. To improve efficiency, CKIN (1)=81 and CKIN (2)=101 was used.

W+jet samples: Processes 16 and 31 were activated and CKIN (3)=100 was used to force the generation of events at high transverse momentum. The W bosons were forced to decay leptonically.

Z+jet samples: MSEL=13 was used with MSTP (43)=2 to turn off contributions from virtual photons. The Z bosons were forced to decay leptonically. (Note that the sample is not identical to the one generated for Data Challenge 0, as the parameters and version of Pythia are different.)

⁹ Pythia Reference

¹⁰

¹¹ A Moraes, I. Dawson and C. Buttar, ATLAS note in preparation

¹² ref to discussions in the MC4LHC group

Photon+jet sample: MSEL=10 was used with CKIN(3)=100.

Inclusive top sample: Processes 81 and 82 were used with MSTP (7)=6 to force the production of top quark final states. The decays are unbiased.

Higgs Samples: Inclusive Higgs production, using processes 102, 123 and 124, was generated for Higgs masses of 120 and 130 GeV/c², respectively. The former is forced to decay to $\gamma\gamma$ and the latter to four leptons which can be either e or μ .

The WH process (26) was used to generate events with Higgs masses of 120 and 400 GeV/c². The W was forced to decay to $\mu\nu$ and the Higgs to one of bb, uu, cc or gg thus making eight sets in all.

A separate ttH production using processes 121 and 122 was made; the H, with a mass of 120 GeV/c², was forced to decay to bb, One W was forced to decay leptonically to e ν or $\mu\nu$ and the other to jets. Only one sign of leptons is generated.

MSSM Higgs Samples: These were generated, using a SUGRA model so that sensible widths are obtained; IMSS (1)=2, RMSS (4)=1, RMSS (5)=39, RMSS (1)=212, RMSS (16)=0 and RMSS (8)=483.4. The H has a total width of 10.3 GeV and a mass of 400 GeV/c². Note that all the H (400) cases correspond to the same model. For a H mass of 300 GeV/c² the following parameters are used: IMSS (1)=2, RMSS (4)=1, RMSS (5)=42, RMSS (1)=144, RMSS (16)=0 and RMSS (8)=436.

Processes 152, 173 and 174* were used to generate the final state $H \rightarrow hh$ and the h is forced to decay, so that samples with bb bb, uu bb are produced for the 400 GeV/c² Mass; only the bb bb state is produced for 300 GeV/c². (3 samples in all)

Processes 181 and 182 with KFPR (121,2)=5 and KFPR (121,2)=5 were used to make the bb H final state; H is forced to decay either to bb or uu for the 400 GeV/c² mass case.

6.2 Generation Monitoring

The quality of the generated events produced was assessed and controlled with the use of histograms of various characteristic properties of those events. These histograms were produced after the generation by running a job that invoked a purpose-written algorithm called HistSample (in the ATLAS ATHENA framework) on the generator output (ROOT-IO format) HistSample produced the basic histograms that were then written to RZ-format output files. An n-tuple was also written into the same file, of which more will be said below.

The 'prototype' sample considered was comprised of Z+jet τ events with the Z decaying to e^+e^- , $\mu^+\mu^-$ and $\tau^+\tau^-$. The HistSample histograms were of quantities including the p_T and the mass of the generated Z as well as some more general quantities such as the rapidity, pseudorapidity and number of charged tracks in the event. In addition, the mass of e^+e^- , $\mu^+\mu^-$ and $\tau^+\tau^-$ pairs and p_T of leptons were histogrammed. Despite being constructed for a specific sample, these histograms are of use for many other event classes, although clearly the reference histograms to which they should be compared will differ.

The n-tuple that was also produced by the HistSample algorithm contains quantities related to the jet structure of the event. Jet finding was performed by running ATLFAST with the normal smearing turned-off, and then making use of the associated ATLFAST utilities to perform the jet finding at the particle level in the generator output. The n-tuple is then used in a secondary job, which runs a KUMAC in the PAW environment to produce histograms of the number of reconstructed jets, their p_T spectra and pseudo-rapidity distributions. These are normalised in various ways: to the number of events; to the number of jets; and to the total cross section. It is because of the need for these latter normalisations that this second set of histograms was produced in a second step and not in the HistSample job.

Finally, the two histogram samples were merged and a postscript summary of all of the histograms produced was made and checked for consistency with the physics expectations for the given sample by human visual inspection; in future, automated comparison with reference histograms will be possible. The various output files were put into long-term storage using CASTOR.

The system encountered various difficulties, notably the access to the very large input files. The Root-IO routines did not seem to be RFIO-aware, which meant that large local copies had to be made. This gave problems both with the available disk space and with the time to stage the data.

6.3 Event Simulation

The ATLAS detector simulation was done in the ATLSIM framework using GEANT3. ATLSIM is a PAW-based framework, which uses KUIP instead of FFREAD for job control. It has an improved memory management, eliminating any hard limits on the track/vertex/hit numbers. It also has improved hadronic physics based mainly on the GCALOR package. A number of known infinite loops were eliminated. To avoid known problems with low energy K_L^0 (zero cross section in FLUKA), they are always traced by GEISHA.

ATLSIM uses plug-in components (shared libraries) to provide extra I/O facility (ROOT, Objectivity) and to load ATLAS detector geometry. Description of the ATLAS geometry is taken from the DICE package. Compared to the Physics TDR, several updates have been made to reflect the design modifications since 1998:

- Beam pipe: multi-layer beam pipe design.
- Pixel detectors: symmetrical, insertable layout
- SCT: tilt angle reversed (to minimize cluster size)
- TRT: modular design of the Barrel
- Inner detector services material updated
- Realistic field in Inner Detector

- All End-Cap Calorimeters shifted by 4 cm
- ENDE: dead material, readout updated
- TILE: material and readout update
- HEND: dead material updated
- FWDC: detailed design with precise rod positions

- Muon system design corresponds to the AMDB version P.03

During the simulation-phase di-jet events produced by PYTHIA were analysed by a filtering routine which looked for a predefined energy deposition in two neighbouring towers in η - ϕ space. (Accidentally, the tower size was selected as $d\eta=6.28/105=0.0598399$ and $d\phi=2*3.80/127=0.598425$). Only events selected by the filter were passed to the simulation step and then written out.

6.4 Quality assurance and data validation

The aim of the ATLAS DC validation¹³ has been to insure the compatibility and reproducibility of the samples produced by different simulation sites. In addition, the validation process has served as a general monitoring tool of changes/improvements to the ATLAS detector geometry. A histogram-by-histogram comparison is performed between two sets of validation histograms, providing a bin-by-bin significance plot and a chi2.

To accomplish this task a semi-automated system to compare and identify differences between two simulations was set-up. The validation test-suite consists of a modular analysis structure based on PAW, which runs off a general purpose n-tuple (CBNT) from the ATLAS reconstruction framework (ATRECON), with information on MC event generation and reconstruction by all ATLAS sub-detectors.

The analysis procedure consists of two steps. First, a (open-ended) list of sub-detector specific macros are run from a master process to produce the validation histograms for the comparison. Thereafter, a histogram-by-histogram comparison is performed between two sets of validation histograms, providing a bin-by-bin significance plot and a chi2. At the end a summary chi2-bar chart for all compared histograms is made.

A broad variety of validation samples of dedicated single particle scans and physics benchmark processes (H, Z) were produced to validate the full simulation chain. This initial phase proved to be important for the discovery of missing/faulty/new aspects of the detector geometry implemented in the simulation. This phase demonstrated

¹³ <http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/DC/Validation/www/>

also, that for the validation of a larger set of samples, the quality and stability of services such as afs, castor or batch system have to be improved, in order to reduce the amount of effort and time needed.

The validation of participating institutions was done by comparing the simulation of identical input samples from different sites and by comparisons of larger, statistically independent, samples of the same physics process. The validation provided an important checking of the simulation infrastructure at the contributing DC sites.

Up to now, only few people participated actively in the data validation. The complexity and intensity of this activity during the initial phase would probably not have allowed to successfully extend the group of people involved much further. However, a stronger engagement of the physics groups in the period, which is now starting, is a critical issue for the validation process.

The physics validation of the DC1 data has already started. The di-jet 17GeV sample (15M events) has been processed with the fast simulation and reconstruction package, AtIfastOO. Comparison of the events content for multiplicities of jets, b-jets, c-jets, electrons and photons, with a similar sample produced in '96/97 large-scale production, is available. In addition fully simulated samples have been inspected and are presently used e.g. for detailed detector calibration purposes.

6.5 Bookkeeping and Database

Essential components required for ATLAS Monte Carlo production are the associated bookkeeping and meta-data services. For DC1 the ATLAS Metadata base Interface, AMI, and the Replica Catalogue, MAGDA, were used to store information such as the logical and physical filenames, as well as a detailed description of the datasets.

6.5.1 The AMI Database ¹⁴

The term "bookkeeping" is used in many different contexts in HEP. Here we use the term to refer to a database application whose purpose is to store data that describes binary physics data that may be either real detector output, or, as in the case of DC1, simulated Monte Carlo data. The application is more precisely described as an "Application Metadata Catalogue". The term "Application" signifies that the information which is stored is specific to the application, and could not be guessed by some outside system, for example Grid software. The aim of the Application Metadata catalogue is twofold:

- to make it possible to understand the contents of a file without actually having to open it,
- to search for a data filename or list of filenames, given a set of attributes of data.

The bookkeeping application used in DC1 is part of a development at the ISN, Grenoble. It was first used as an online electronic notebook for LAr test-beam data, and later adapted at the ISN as a prototype application to store the metadata of offline calculations. This prototype was used in DC0, without much success, although considerable effort had gone into refining the list of metadata attributes to be stored.

For DC1, the first priorities were:

- a command line interface for input of metadata,
- a user manual,
- a prototype of a web searching interface.

Database Design

The bookkeeping application uses a layered architecture. It is written in JAVA, and makes use of the generic JDBC library for SQL database communication. In consequence, it is independent of platform, operating system and database technology. The only prerequisite is that java is installed on the client system. The architecture allows for geographic distribution of bookkeeping; all connections pass through a central router, which redirects requests to the correct site. The central router should be mirrored. For DC1 however, all the databases are physically at the ISN Grenoble, and are situated on the same server.

The core packages manage the remote connection to the database, and the transmission of SQL commands. This means that we can use any database which understands SQL, and for which a java JDBC driver is available. The

¹⁴ <http://larbookkeeping.in2p3.fr/AMI/>

middle layers provide generic classes for accessing the bookkeeping databases, using their internal descriptions. Top layers of the software are provided for particular interfaces, such as the command line interface, and the web interface, or even for separate projects. The application is called the "ATLAS Metadata base Interface" (AMI).

The Command Line Interface

The authors first provided a document¹⁵, which describes the proposed specification of the interface. The proposal was discussed with members of the production team. In the light of discussions, the document was modified, and agreement was reached on the priorities for the first version. The first version of the AMI command line interface was delivered at the end of May 2002, and a second enhanced version was released at the end of July 2002

The Prototype Web Interface

A web interface that will allow users to search the bookkeeping databases has been provided. The main difficulty in establishing such an interface is the lack of use cases from physicists. It is hoped that the prototype will trigger user reaction so that useful development can continue.

Future Development Plans

In the light of our bookkeeping experience with DC1 phase 0, it appears necessary to revisit and refine the set of attributes used to describe the simulation step of production, and also to establish what attributes should be stored by the bookkeeping to describe other processing steps, such as reconstruction. As described above, we will continue to develop both the command line interface and the web interface, as we receive user feedback. The ATLAS Metadata base will be interfaced to the EU Grid WP2 package "Spitfire"¹⁶. This package provides a secure grid-enabled front-end to relational databases.

A bookkeeping requirements document¹⁷ is in preparation, and should be reviewed in the autumn. After this step, a formal design document can be prepared.

6.5.2 MAGDA

Magda¹⁸ is being developed to fulfil the principal ATLAS '01-'02 deliverable for the Particle Physics Data Grid (PPDG) project of a production distributed data management system deployed to users and serving BNL, CERN, and many US ATLAS grid test-bed sites.

Magda makes use of the MySQL open source relational database, Perl, Java, and C++. MySQL was chosen because of an existing experience base with it and because of its proven performance in data management for the STAR/RHIC experiment. For data movement gridftp and (where grid infrastructure is not available or usable) bbftp, scp are used. The Globus replica catalogue is currently being integrated.

The 'core' of the system is a MySQL database, but the bulk of the system is in a surrounding infrastructure for setting up and managing distributed sites with associated data locations, data store locations within those sites, and the hosts on which data-gathering servers and user applications run; gathering data from the various sorts of data stores; interfacing to users via web interfaces for presenting and querying catalog info and for modifying the system; and replicating and serving files to production and end-user applications.

All the 2000 files generated for dc1 in the U.S. Grid test-bed were put in the BNL HPSS storage system using Magda. Magda also managed the replica location for these files – so all 4000 files were automatically registered in Magda.

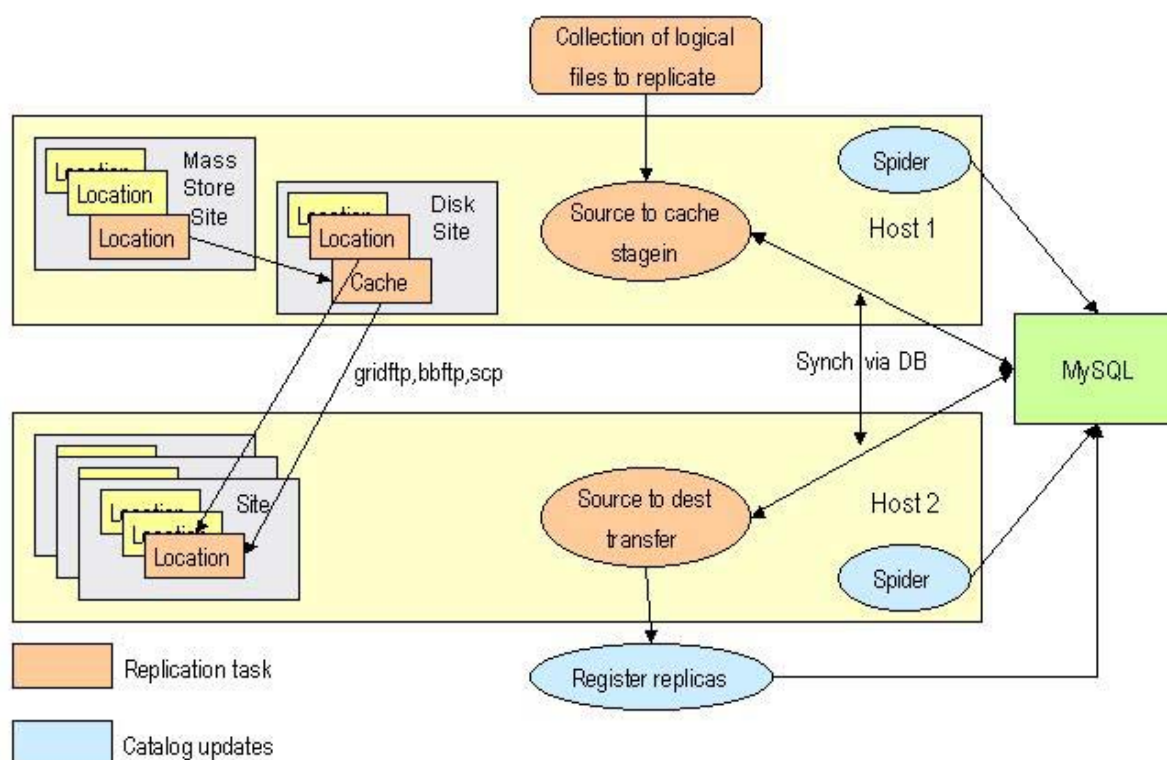
¹⁵ <http://larbookkeeping.in2p3.fr/AMI/ExternalSpec.pdf>

¹⁶ <http://hep-proj-spitfire.web.cern.ch/hep-proj-spitfire/server/doc/index.html>

¹⁷ http://larbookkeeping.in2p3.fr/AMI/BookkeepingRequirements_1.pdf

¹⁸ <http://www.atlasgrid.bnl.gov/magda/info>

Magda Cataloging and Replication Architecture



6.5.3 Prototyping Virtual Data Approach

Because of the physics-oriented content of ATLAS Data Challenges the recipes for producing the ATLAS data (ATHENA jobOptions and other similar "input data cards" files) have to be fully tested. The data produced have to be validated through a subsequent quality assurance and validation step. Preparation of the production recipes takes time and efforts, encapsulating considerable knowledge inside. Due to a smaller scope of ATLAS DC0 more time has been spent to assemble the proper recipes than to run the production jobs. Having the proper recipes, producing the data is straightforward. Because of the prevailing vision that the data are primary and the recipes are secondary (they needed just for the data production) it has not been clear how to treat the developed recipes after the data have been produced. It was decided to store these recipes outside of the scope of the ATLAS Bookkeeping Database AMI.

A valuable insight for ATLAS production workflow has been provided by introduction the virtual data concept. The GriPhyN project¹⁹ provides a different perspective:

- recipes are as valuable as the data,
- production recipes are the virtual data.

Taking this approach to the extreme means that if you have the recipes you do not need the data (because you can reproduce them), i.e., the recipes are primary and the data are secondary. According to the virtual data architecture, recipes are stored in the virtual data catalogue database.

In the process of the ATLAS Data Challenge we have evaluated the virtual data approach for the production of several datasets. The ATLAS database group developed and delivered an infrastructure for early application of virtual data concepts and techniques to ATLAS data production. A virtual data catalogue database prototype was

¹⁹ <http://www.griphyn.org>

deployed in the spring of 2002 for evaluation in the context of the ATLAS Data Challenges. The prototype is being used successfully for data challenge event generation and detector simulation. Production job options for physics event generation and production scripts for detector simulation were recast as parameterised transformations to be catalogued, with the resulting parameterisations represented as derivations. ATLAS DC0 and DC1 parameter settings for simulations are recorded in the virtual data catalogue database.

The production system, based on the virtual data catalogue prototype, implemented the scatter-gather data processing architecture to enable high-throughput computing. The production fault tolerance has been enhanced by the use of the independent computing agents, adoption of the pull-model for agent tasks assignment (instead of push model typically used in batch production) and by the local caching of output and input data. An interesting feature provided by this architecture is the possibility for the automatic "garbage collection" in the job planner in the following sequence: production agents pull the next derivation from the virtual data catalogue; after the data has been materialized, agents register "success" in the database; when previous invocation has not been completed within the specified timeout period, it can be invoked again.

6.6 Error messages during event simulation

In this section we summarize the different error messages during the event simulation with comments from Pavel Nevski.

GCALOR problems:

CALOR: Fatal ERROR in EVAP and
CALOR: ZEBRA banks screwed up --> STOP

- bugs in GCALOR. Due to the lack of accuracy sometimes the energy in the evaporation model is not strictly conserved at an MeV scale. This happens rarely (in less than 10^{-4} events) and is rather invisible in the range of typical hadronic energies in ATLAS. In the future this error should be ignored in the code.

Suggested solution to by-pass STOP:

:

Program has to be restarted with a different random number seed: RANLUX \backslash \$sigma(\$OUTPARTNR + 1000000)

Other GCALOR non-fatal error messages:

GUSTEP ERROR: after 1 iterations and 0 particles done

wrong handshaking between GCALOR and GUSTEP when more than 100 secondary particles are produced in a single hadronic interaction. . Will be corrected later, $< \sim 1\%$ events (but several lines printed). Actually this may produce a single hit with a huge energy (overflow) and should be taken into account later in the reconstruction.

ERROR GCALOR: Particle type 1380927008 not implemented in GEANT

- gcalor loosing the particle id, $< 10^{-3}$ events affected. This is again caused by a wrong handshaking between GCALOR and GUSTEP. When more than 100 secondary particles are produced in a single hadronic interaction, some of the particles from the above excess can be lost.

*** Strangeness non conservation in Hadriv -1 15 8 ***

PROJECTILE HADRON MOMENTUM OUTSIDE OF THE ALLOWED REGION, PLAB= 0.90575E-05

- precision problems in GCALOR, should be ignored

Ferevv: Umo2 < Urmin2 !! 6.03708507 7.37604129 20 1 1.19743 0.93827231

- precision problems in GCALOR, should be ignored

I/O system problems:

1. ***** event loop ends because the IQUEST flag set by program is -1
- input error; input ROOT file is corrupted; input file should be re-copied or re-created.
2. Error: cannot open file "iostream" [FILE:/tmp/filebCTcOM_cint](#) LINE:2

*** Interpreter error recovered *** - problem in the local ROOT installation (wrong or missing system.rootc file), can be ignored
3. error in CFPUT : Can't open configuration file

- This and other CFPUT problems appear when a local output file cannot be written (due to lack of space, denied access etc.). Job should be re-run

Harmless warnings

1. GSNCTR ERROR - I,K = 2 2 SN = 0.86949E+03, etc

Old known problem with accuracy of a helice crossing 2nd order surface in spanish fan. The message is a trace back of the improvement done a while ago by Sasha Shehtman - he reduced inaccuracy the tracking down to an acceptable (few micron) level. No improvement is planned; seen in every run

2. ***** ERROR in HNORMA : Unknown histogram : ID= 764

Leftover message from a developer control, no danger, seen in every run; should be removed by Serguei Baranov later.

3. MDTDIG WARNING! Digitisation Overflow Nvl are 2 6 48 2** 0

MDT digitisation, $< 10^{-4}$ events affected, (but MANY lines are printed); a looper produces too many hits in a tube. This problem will be corrected later during re-digitisation with a later version of the muon code.

4. PIXBDIG WARNING! Digitization Overflow (Layer,Sublayer,Iphi) 1 20 2 11 *

- some pixel digits are lost in the readout buffer in a highly occupied wafer. Actually same loss will happen in reality because the readout buffer will have a similar depth.

5. Aucun strip touche ! Nstrips = 0 steps = 21

A hit happened in LAr in a non-sensitive area. This is not an ERROR message.

7.) Pile Up

7.1 Pile-Up Procedure

The cross-section for inelastic, non-diffractive pp interactions at the LHC is expected to be around 67 mb. At design luminosity ($10^{34} \text{ cm}^{-2}\text{s}^{-1}$), the average number of minimum-bias events is 23 per bunch crossing. This number varies according to a Poisson distribution. Pile-up is added on the hits level for all detectors according to their 'memory' time that varies between one bunch-crossing in silicon detectors and 30 bunch-crossings in the calorimeter and muon systems. For this purpose a new highly efficient event mixing procedure was implemented. In the muon system additional pile-up is arising from the cavern background. To take care of this

effect, special minimum-bias files will be produced which include the cavern background on top of the “normal” pile-up event.

To generate signal events in the presence of pile-up, minimum-bias events are added on top of signal events using the ATLSIM framework. The number of bunch crossings “seen” by a sub-detector is specified individually. In the inner detector one bunch crossing is seen by the pixel and SCT detectors, whereas the TRT is sensitive to two bunch crossings prior and after the main bunch crossing respectively. A method, which uses the flight time, needs to be invoked in the TRT digitisation routine. In the LAr calorimeters the bipolar shaping functions have a response, which lasts for over 500ns. The Tile calorimeter is sensitive to two bunch-crossings prior and one after the main bunch crossing. In the digitisation routines the response to the bi-polar or uni-polar shaping functions have to be taken care of. In the muon system the “normal” pile-up from ~ 25 bunch crossings needs to be considered. In addition, pile-up coming from the cavern background needs to be added.

To generate signal events for a given luminosity in ATLSIM the signal and the minimum bias event files need to be provided. For bunch crossings -2 to 2 pile-up is added on the hits level for all detectors directly from the second (minimum bias) input event stream. The pile-up for earlier bunch-crossings as seen by the calorimeters and the muon system are added using pile-up events stored in a “huge bank”,. This is a special event store, optimally organised to minimise system swapping. The bank resides in the system swap space and only its parts are brought into real memory on request. Interaction time of the pile-up events is randomised on every new physics trigger, so that with the same set of minimum bias events in the “huge bank” one can generate many different pile-up signal shapes. After each event ~10% of the minimum-bias events stored in the “huge bank” will be replaced by new events. The thus generated events are stored on file. Compared to the input tapes no hits information is stored due to disk space considerations. The events have the same run and event number as the original signal events.

Compared with the pile-up method used to produce the physics TDR tapes, pile-up is added in the muon system for DC1 for the whole production. In 1997 the muon system was not simulated for the di-jet events. Per bunch crossing 24 minimum bias events were added instead of the “nominal” 23. In the “old” pile-up method the full correlations within the hits in the different detectors were only considered for the main bunch crossing. In the TRT 32 minimum bias events were added at bunch crossing 0, which approximates the effect of minimum bias events coming from other bunch crossings in the TRT. In the calorimeters pile-up was added using the correct shaping functions. To do so, the pile-up not arising from the main bunch crossing was added at DIGI level on top of the calorimeter energy matrices (as used in ATRECON) within ATLSIM.

NEW TEXT FROM PAVEL ON PILE-UP

Full ATLAS pile-up simulations.

Any collision registered in the ATLAS detector, contains in fact a superposition of particles coming from a single "physics" event, which triggered the readout, and of particles coming from another un-selected pp collisions.

On average per every bunch crossing seen by the ATLAS detector or a subsystem one expects to have 23 "unbiased" overlaying collisions. The total number of observed particle per event depends on the signal collection time, which varies from few ns in silicon detectors to about 700 ns in the MDTs.

In addition, while in LAr calorimeters the signal is measured shortly after the trigger so that it is affected only by previous bunch crossings, measurements in drift detector such as TRT or MDT continues for the maximum signal collection time so that they are sensitive to the same amount of posterior bunch crossings.

The full pile-up is simulated as a number of minimum bias collisions properly distributed in time and overlaying the physics collision.

As every collision is normally simulated only for few 100 ns of the propagation time, there is one additional component missing in this scheme: Neutrons may fly in the ATLAS cavern for few second until they are thermalised, thus producing kind of a permanent neutron-photon gas which creates a constant rate of compton electron and spallation protons observed in the muon system. This component, i.e. additional hits created by long living particles, is called "cavern background".

Cavern background is simulated as a separate component that is added on top of every single minimum bias event. This is done in the following steps:

- 1.) A standalone dedicated GEANT3/GCALOR based detector simulation program with improved neutron propagation and a simplified ATLAS geometry is run on pp collisions (by Mike Shupe). The output of this program provides particle fluxes in the envelopes surrounding muon chambers. The fluxes are provided as list of particles with all related parameters per a pp interaction on the entrance of each chamber envelope.
- 2.) ATLSIM randomly reads from these fluxes an average number of particles per single pp collision and feeds a subset of them into ATLAS DICE geometry. At this moment all photons and neutrons entering the chamber envelopes are selected ($E_{kin} > 10$ KeV). Charge particles are selected only the first time they appear in the output list and only if their production time is bigger than the time cut-of of the DICE simulation, so that the prompt component of the calorimeter punch-through is not double counted. The starting time of all selected particles is reset to 0-25 ns interval.

A significant randomization is achieved at this moment due to:

- a.) random initial particle selection;
- b.) low probability of neutron and photon interaction in the chamber envelopes;
- c.) arbitrary selected particle rotation at the input

This allows multiple re-use of the particle fluxes simulated in the first, the most CPU consuming step.

The detailed muon system geometry description provided by DICE is used to simulate signals induced by the cavern particles in the muon chambers.

The initially selected neutral particles are propagated only within chamber envelopes to avoid double counting of the n-gamma cascade. However, all their products and initially selected charged particles are trace until the GEANT program stops them.

Hits produced during the tracking (usually in the same 0-25 ns time range) are saved in pseudo-events normalised per one pp collisions as a standard (ATLSIM) simulation output.

- 3.) Output of the cavern background simulations is mixed with the standard fully simulated minimum bias events (dataset 2099), thus producing new minimum bias events with the cavern backgrounds included. Mixing proportion may varies from 1 to 10 as the "safety factor" requested by the Radiation Task Force. (K^0 and their decay product are already correctly simulated to some extend in the normal minimum-bias tapes as ATLSIM contains the known bug correction for the K^0 propagation) This approach drastically reduces the time need to simulate the signals induced in the muon spectrometer by the cavern background comparing to the previously used technique. In the same time it allows for a realistic compton electron and spallation proton production, which takes into account, all geometry details available in DICE properly convoluted with dedicated the n-gamma fluxes calculations.
- 4.) The resulting minimum-bias events should be added as a pile-up to any physics events. This should be done taken into account the LHC luminosity and bunch structure. To fully simulate the complete detector pile-up mixing should be done for +/- 30 bunch crossings (in the same way as it was done for the inner detector for the Physics TDR) with the average number varying from 4.6 events per bunch crossing for the low luminosity ($L=2*10^{33}$) run to 23 events per bunch crossing for the high luminosity ($L=10^{34}$) run.

CPU and memory requirements:

Step (1) is made once for a muon system layout by Mike Shupe. We need about 10K simulated events, which is only a small fraction of regular flux calculations. However, each simulated event takes at least

Step (2) is also done once by a special version of ATLSIM with the standard DICE geometry taken from the production release 3.2.1. This step takes about 0.3 sec per simulated event and requires standard ATLSIM memory (<100 MB per job). The output is produced in files that contain 10K event (for comparison, Data set 2099 has 500 events per file). This is more than is needed for one to one file mixing at any reasonable safety factor. The total number of events needed at this step is about 1000 files * 10K events - 10M events, the simulations time - $3*10^6$ CPU secs.

Step (3) should be done several times per each minimum bias tape. (for every selected safety factor 1,2,5 as planned for the moment). As each job requires one input file from the dataset 2099 and one cavern background file, all three mixing could be done in one job. Each such job requires less than 100 CPU-secs but is output extensive (each 300MB input file yields 3 files close to one GB in total).

All together 1000 pre-mixing jobs are needed. The resulting files should be distributed over the production sites involved in the physics pileup production. (If a big enough temporary disc storage is locally available, it is possible to make this step "in flight" as a part of step 4.)

Step (4) is the most time consuming procedure as in addition to the event mixing it requires running full digitisation of the ATLAS detector. Time require per job does not depend on physics but on the luminosity only. A high luminosity pile-up job requires a 500 MB machine and takes 220 CPU-secs - 40 secs for mixing, 180 secs for digitisation.

This step produces output events of about ~ 8 MB at high luminosity independent on the input physics event size.

The memory requirement (500 MB) is a matter of concern as most of the CERN machines have only 256 MB. This will be improved with the next release ($>5.3.0$).

7.2 Resources needed for Pile-up production

As for the standard ATLSIM/Geant3 simulation the digitisation is accounted for in the simulation. We estimate the following average numbers for piled-up events in the $|\eta| < 3$:

Luminosity	Output size/event (MB)	CPU-time/event (SI95sec)
$2 \cdot 10^{33} \text{ cm}^{-2} \text{ s}^{-1}$	3.6	2000
$10^{34} \text{ cm}^{-2} \text{ s}^{-1}$	7.5	8000

The list of runs be processed can be found on the web²⁰.

²⁰

http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/DC/DC1/DC1_2/production_requirements.html

8.) Data Challenge 1 and the Grid

Recent advances in computing can be characterised by emerging Grid technologies. Powered by various middleware, Grid computing infrastructures are becoming a reality, and as such are particularly important for large distributed projects like the High Energy Physics experiments, and ATLAS in particular. By harnessing distributed and scarce resources into a powerful system, the Grid is expected to play a major role in a not-so-distant future. Apart of optimization of the distributed resources usage, the Grid will naturally offer all the collaboration members a uniform way of carrying out computing tasks. This is essential for large production tasks, which need plenty of resources, both hardware and human, worldwide.

Data Challenges are the perfect opportunity to evaluate the current status of the Grid middleware and assess what has to be done by the collaboration in order to make a smooth transition to the Grid tools. Therefore ATLAS has been extremely active in Grid matters over the last few months.

8.1 Grid activities in Phase 1

A significant fraction of the first phase of the ATLAS DC1 was performed in the Grid environment, involving 11 out of 39 sites.

Of the Dataset 2000, 15 input partitions were processed by the NorduGrid²¹, and the whole Dataset 2003. The test-bed included 8 Linux clusters across the Scandinavia. Despite having different operating systems and hardware characteristics, the clusters performed as a single farm, having jobs distributed in optimal way, and writing the output onto a dedicated storage area at the Oslo University. A detailed report can be found on the web²².

Three sites of the US Grid test-bed took part in DC1 phase 1. There a significant fraction of the 1 TeV single particle production was done.

8.2 Further ATLAS involvement in Grid activities

In the next phase of DC1 (October-December) the production could be done as before. However, we would like to increase the usage of the Grid. Here inter-operability among Grid projects would be an asset, but it is not a condition sine qua non.

The final phase of production (January-March 2003) will be performed with jobs running at those sites where the simulated and pile-upped events have been produced and stored. The amount of data to be accessed by the reconstruction will be about 50 TB. The use of Grid tools will allow for easy access to all sites, to perform reconstruction of any sample regardless of its location. Without Grid tools this production would still be feasible, however, only by asking the production managers on each site to run the wanted jobs.

ATLAS is committed to work very closely with the LCG project. Therefore, these tests are foreseen to be performed in close collaboration with the LCG management and other already planned initiatives and projects (e.g. DataTag). We would fully support all common efforts of the LHC experiments and bi-lateral initiatives between the Grid projects.

Experiment-driven activities with very well defined goals will help the LCG project in their choice for a common set of middleware to be used for the first LCG Grid service, due later in 2003.

²¹ <http://www.nordugrid.org>

²² <http://www.nordugrid.org/documens/ATLASdc1.html>

8.2.1 Grid test in preparation for Phase 2

In order to evaluate the possible usage of the European DataGrid (EDG) tools for DC1 phase 2 and to provide the necessary feedback, a task force was set up with members from both ATLAS and EDG. The task of this group was to repeat a fraction of the DC1 phase 1 production on the EDG test-bed. The sites involved in this test are the so-called "core" EDG clusters (CERN, RAL, NIKHEF, CNAF, CC-IN2P3) plus one cluster from outside EDG (GridKa Karlsruhe). A status report is being drafted²³.

8.2.2 Possible Tests for November

Since ATLAS is interested in promoting inter-operability among Grid projects, a limited set of requirements was identified that would be useful for the next steps. These requirements were circulated to several potential Grid 'providers' (e.g. EDG, NorduGrid, US ATLAS Grid), culminating in a half-day session on September 19 at the recent ATLAS Software Workshop. It was agreed that ATLAS should proceed, very rapidly, to specify in more detail its requirements, in terms of a test using a real ATLAS application. These requirements should then be passed to the Grid projects for technical evaluation. If found to be feasible, this test, using the "pile-up" application, should be carried out in the second half of November. The results of the November test will inform ATLAS' use of Grid tools for the large-scale operations next year.

As a step towards the goal to reach some inter-operability between Grid projects we intend to perform the following tests (in increasing complexity) in November after the demonstrations planned by DataTag and the US Grid project.

- 1.) **Extend the EDG test-bed to more ATLAS sites** (sites which are members of DataGrid, CrossGrid or are already recognized by EDG). Including also some non-European sites would be a relevant asset. The amount of CPU and storage resources at each site is not an important issue for the tests in November.
- 2.) **Test a basic implementation of a worldwide grid:** Use a central production DB repository, and the existing job schedulers, EDG, NorduGrid and US. The tasks to be performed are: access the production DB, select for each job the site where it should run, mark the job as in progress, run the job, register it in the data catalogue, and then mark the job as done in the production DB and take the next available. Part of these tasks can be performed by the different Grid schedulers and part by ATLAS scripts and interfaces. Job submission should come from well-established locations that are registered in a central ATLAS Virtual Organisation (VO).
- 3.) **Select and employ a single resource broker and job submission protocol that employs all the available resources defined as the ATLAS grid.** (These types of tests are currently performed as part of the DataTag/GLUE project.)
- 4.) **Achieve a definition of components that must be used as part of a global release, based on experience with the above test, to define a standard package for all ATLAS users in a Grid environment.** Completely identical configurations and releases may be quite tricky, especially given the different patches to GLOBUS - but it is conceivable that a widely agreed upon patched GLOBUS release may be workable.

How far we can actually go we will learn after having discussed with the technical experts and all the parties involved. On the other hand we may profit from the fact that other parties have performed part of these tests.

In summary, there should be coherent effort that involves the LCG, the grid-developers and the LHC experiments, in order to reach the final goal: THE common Grid middleware deployed by the LCG project. The proposal outlined in this note is intended to contribute to that goal, and is fully consistent and consonant with the ideas presented by the LCG project leader to the LHCC on September 30, 2002..

²³ ATLAS Data Challenges on the EDG Testbed 1.2 (paper in preparation)

9.) Production Reports From Different Sites

9.1 Australia

a.) Software installation:

Prior to DC1 Phase 1, we were trying to install the software via the CERN /afs tree and this was extremely painful and time-consuming since there appeared to be so many assumptions made that the software was running on a CERN node and not at a remote site. In addition it wasn't clear which parts of the tree we needed to copy so we ended up copying a lot more than necessary which was very slow given the speed of our international links from Australia. And there was a complete lack of basic documentation explaining how to install and run the software.

However, in time the RPMs from INFN became available and after that installation became far, far easier for us. We run the CERN release of RedHat Linux on our cluster since we thought that would make things easier for us in the long run. The RPMs install onto this with no difficulty in a very short amount of time, and since we make heavy use of NFS we only needed to install the RPMs on one node and the software was then instantly available to all nodes in our cluster.

Once the RPMs were installed the biggest hurdle was the front-end or "prodscrip" as it's known. This still required considerable customisation since the default one is very CERN-centric. I also found it a bit frustrating that the prodscrip was not really part of the "release" - even though it's an absolutely vital component. A version of this script was provided with the RPMs, but there was also a version available from the CERN /afs tree in someone's home directory (no less) with an obscure name like "prodscrip4" which most people seemed to be using! I found this very confusing and wasn't sure which one should be used. The only reference on the DC1 details screen simply said "prodscrip needs updating ...".

Basic documentation was still a little lacking. The DC1 "details" page was extremely useful but needed to be more comprehensive and tackle things from start to finish. A physicist working at CERN can get most of what he needs from it to set-up-set-up and run the simulation, but if you are not working at CERN and/or you're a computer scientist then it would be very, very hard to set-up and run the simulation. I suppose the hope is that use of the grid for Phase 2 will largely solve this problem?

b.) Problems during the data processing:

Most of our problems were the same as those experienced at other sites I imagine - the bugs discovered along the way required us to reinstall software and restart the simulation from scratch. One job failed due to the random number problem and had to be restarted with a new seed. Other jobs needed a restart because I made an error editing the KUMAC in the prodscrip.

The only other problem then was just monitoring the jobs for success or failure and resubmitting where required. prodscrip does not seem to exit with a failure code when things go wrong so the only way to be sure the job worked is to eyeball the logfiles and there can be quite a lot of those!

c.) Problems getting data to/from CERN

No real problems as such here. We made heavy use of bbftp which sped things up quite a bit. Since we are at the end of a (relatively) slow international link it took quite a while to send/receive data. Uploading results typically took 1-2 days per dataset (we were seeing around 600KB/sec xfer rate with bbftp to castor).

Sometimes transfers would fail part way through but bbftp will auto-restart from where it left off and resume so that wasn't a problem.

One comment I will make on this point however is that I think that in future perhaps it would be good to provide checksums/md5s of input files (either on a web page somewhere or alongside the file on castor with a .sum or .md5 extension). During one early trial run we spent quite some time trying to figure out why our job kept

crashing only to discover we had a corrupt input file. A simple checksum comparison would have saved us a lot of time in this case.

9.2 Canada

a) Software installation

For the DC1 Phase 1 the ATLAS software was installed on the two main Alberta clusters using CMT and the cvsupd daemon to download the required software packages for the release.

This procedure was successfully started some time ago with release 3.0.0 and included all releases up to and including 3.2.1. Future releases will also likely be installed in this fashion. I find this method more flexible than the RPM based method. The goal of the installation at Alberta with cvsupd is to provide the user with an environment which allows for code development in a fashion similar to what is found at CERN.

The RPM method was used to install the ATLAS software on two other clusters at the University of Alberta to which I did not have root access. The fact that the RPM files required root access to be installed was a problem. Furthermore, the RPM files were not relocatable to anywhere other than /opt.

Therefore the software installation using the RPM files proceeded by first unpacking the RPM files onto a system, which I controlled, and subsequently creating a tar file to distribute to the other machines. With the tar file I could place the ATLAS software where needed without root intervention.

b.) Problems during the data processing:

Difficulties encountered included the occasional occurrence of CALOR errors requiring rerunning with a different seed. Overall the number of problems encountered was small, most of which being hardware in nature.

On a couple of occasions difficulties were had with one of the raid servers used (failed disk drive causing system reset, and then having to be replaced, kernel panic due to Ethernet interface troubles). Two brief slowdowns in production also occurred in order to add and then remove a temporary RAID array to one server used for testing. Otherwise productions have gone smoothly.

c.) Problems getting data to/from CERN

All remote access to CERN for the DC1 productions was through wacdr.cern.ch to download the necessary EVGEN partition files. This was done early on in the DC1 Phase 1 period. At that time only regular ftp was used for the downloads. Protocols such as bbftp will be used for future downloads.

Other than the files from dataset 002000 for validation all files are stored locally at Alberta and were not shipped to CERN.

9.3 CERN

CERN DC1 experience for non-single particles production

After a period of tests the production started for real on Friday July 12. This report covers the period up to Friday August 23, i.e. 43 days.

Allocated computing capacity

The original plan was to run the production over a period of three weeks on a set of 200 dedicated CPUs (part of the LXSHARE cluster). However, during spring CERN-IT decided to change its cluster deployment model from one with many sub-clusters dedicated to single experiments, to one with a single cluster shared by all experiments. At the same time the scheduling scheme used by the LSF scheduler would be changed from 'first come first served' combined with priorities/pre-emption, to a fair-share scheme.

In theory the new plan should have been simple. Given a cluster of about 1500 CPUs, during a period of three weeks ATLAS' share should be increased with a percentage equivalent to 200 CPUs (13%). Reality was quite different.

To start with ATLAS's share was not adjusted until beginning of August. Fortunately, the shares only come into the game when the cluster is full. This was the case only during one out of five days (rough estimate). The other days the production capacity was limited by an artificial 400 job (i.e. 266 CPU) limit, preventing ATLAS to grab all free processors and hold them for a longer period. As a consequence, during the first weeks of production our capacity was going up and down between 50 and 400 jobs, depending on the overall load on the cluster. This was quite annoying especially because at that time we did not understand the reason why this happened.

During the first week of production CERN-IT upgraded their scheduling software from LSF 3 to LSF 4. In this transition period ATLAS had access to two parallel clusters with production sized quota on both. As a result, we were able to run as much as 700 jobs in parallel during a short period.

Two other consequences of abandoning the dedicated cluster scheme are worth mentioning. Firstly, as the production jobs were running on machines shared with other users, they occasionally failed because of misbehaviour (memory or disk space) of these other users. Secondly, the slowest job could take up to three times as long as the fastest job. This is the combined effect of a factor two in raw speed and a factor 1.5 depending upon whether your job needs to share the CPU with a third job or not.

Given that, due to a high failure rate discussed in the next section, often up to three re-submission rounds were needed to finish a large group of related jobs (a dataset), the production of every dataset was spread out over a much longer period in time than expected, leading to a situation that when the production of a new dataset was started, more than 5 others were still not finished.

In hindsight it would have probably been much easier to do this production on a dedicated cluster. On the other hand, on average we managed to use close to 200 CPUs during six weeks, i.e. we did practically twice what we planned to do. Another major advantage is that when for some reason we could not have jobs running, the CPU resources were not necessarily lost. This was in fact one of the main motivations for introducing the new cluster deployment model in the first place.

Experienced problems

The time period over which the production took place, especially the first half of it, was probably the worst possible time of the year. During the first three weeks there was one major (in the sense that hundreds of jobs failed all over the cluster) incident during every weekend and one during every working week: AFS token problems, AFS file servers problems, general network problems etc. Failure rates peaked up to 40% and averaged at 20%. Earlier test productions enjoyed failure rates as low as 0% and consequently our production machinery did not foresee extensive automatic recovery. The manual recovery and cleaning up soon became a full time job for one person.

While 75% of the job failures happened at concentrated times (the major incidents), there were still 25% of continuous failures. Two changes were made to the production scripts in an attempt to remove these. The first version of the simulation script read its input directly from the CERN Castor mass storage system. It was suggested that this was the cause of the many random failures and consequently the script was modified to first copy its input to local disk. This did remove the few errors that were clearly related to time-outs on input but it did not seem to change the much higher rate of segmentation faults, bus errors, abrupt terminations, etc.

Triggered by the observation that in case of network/AFS problems many jobs fail in the middle of computation, whereas in principle they should be totally independent of the network at that time, a small investigation showed that indeed the running jobs have several tens of files on AFS open during execution. Besides the core executable and a limited number of shared libraries about 75% of these files were related to the ROOT package.

Modifying the script to first copy its executable and shared libraries and perform a complete local installation of the ROOT package, made failure rates drop to less than 2%. It is possible that the removed AFS dependencies were the cause of the high failure rates, but part of the improvement could also be due to a decreased incident rate on the cluster in general, which seems to be the case as well. Conclusive evidence would require deploying the two versions of the script at the same time. This does not seem to be worth the effort at present.

Moving from reading the input from Castor directly, to making a local copy first, introduced a potential IO problem. When 400 jobs start simultaneously by copying a 2GB file one might expect problems. Given our

many other problems, we did not even attempt to test this. In practice only the first few datasets had 2GB input files, all others had more modest sizes up to 400MB. Additionally, we took care not to start too many readers of the same file at the same time (<50) and not to start too many readers in total at the same time (<200). With these precautions there was no significant contribution to the overall failure rate. We did not test whether these precautions were indeed necessary, or whether higher limits could have been used.

We conclude this section with a few words on the generator production:

Although all events were generated at CERN, not all of them were generated the same way. Two different production strategies were deployed. We report only on the second one used for all but 4 very high statistics datasets. Because the generator software runs within the ATLAS control framework (Athena) running jobs in batch mode is quite cumbersome. The current procedure for running ATHENA jobs requires you to check out a special Test-Release package from CVS and 'install' it relying upon the ATLAS release tool CMT. This installation sets up your environment variables, creates a multitude of links to executables and shared libraries, and copies a multitude of files locally from the release. In principle, all this can be done from a batch job as well but this is clearly less than ideal.

An alternative strategy is to perform this installation step once in some shared file space and have it reused by the many batch jobs (this is the approach used in strategy one). The approach we exercised requires no CMT based installation phase at all. Instead, one script does everything using the parts it needs from the release directly. Figuring out what these parts were was not easy, as the out-of-the-box set-up links to just about everything in the complete ATLAS release and to many things even three or four times. The script reduces the links from several hundreds to about seventy, some of those probably not needed. One obvious disadvantage of this trial and error approach is that the script can only run the jobs it was intended to run, and perhaps not even all of those.

It is extremely desirable that ATLAS invests some effort in providing reasonable installation support for its software suite, and that software authors specify exactly what their software needs to run instead of using a blind wildcard.

Single Particle and GENZ/Legacy production

A script was prepared to run the SPGUN (single particle gun) work in the zshell as the example script for the first job was in zsh. Then a job submission script was written to process the "one line per sample" file to generate however many jobs were needed. More or less the same was done for the GENZ/legacy jobs.

All of the bookkeeping needs are completed in the job (or at job submission for the information common to each partition in a dataset). There are at least two ways to keep information, so AMI and Magda files are written and "cat-ed" into the log, there are some recovery attempts for copy from the batch job to castor and so on. At CERN MAGDA just registers the file. They are not yet processed.

The job submission logs each job as it is submitted and gives it a unique id appended to the log file name. In the event that the job actually runs and the log file can be found, where it was requested, in \$HOME/LSFJOB... or on the mail server the log file can be inspected and moved into the medium term AFS location before eventual tar archive of the whole directory for that run.

Both the dataset id and the partition number were used for each job to generate the random seed and the parameter string for the script is stored for AMI. A further parameter was added, which is a further digit to add to the seed generation, facilitating a retry for jobs failing with code bugs rather than system errors.

For the legacy GENZ data an appropriate job/script was produced and there is a "pre-processing step".

<http://soneale.home.cern.ch/s/soneale/www/GROUPS/SOFTWARE/DOCUMENTS/UTILITIES/atlsffan.html>
<http://soneale.home.cern.ch/s/soneale/www/GROUPS/SOFTWARE/DOCUMENTS/UTILITIES/atlsfcac.html>

The atlsffan program was used which takes (say) a GENZ file and divides it up into "job size" files and sets sensible run event numbers in all locations - except for the GENZ bank. By setting a very large number of events per partition the complete file can be processed and the output is stored in castor with our newly allocated run number and our conventions for (logical) file names.

9.4 France

a) Software installation

The ATLAS software was installed on the Linux CCin2p3 clusters using CMT and the cvsupd daemon to download the required software packages for the release²⁴. Part of the production (b-tag data samples) was done using the VDC (Virtual Data Catalogue) for storing and retrieving the production job options. The VDC was based on the NOVA MySQL database operated at CERN site.

Some difficulties were encountered due to the lack of information on the needed external libraries. There is no standard ATLAS procedure to deal with external libraries and ONE IS URGENTLY NEEDED. It will be important to publish the list of changes from a release to another one (external libraries like BOOST, ROOT, ANAPHE, CLHEP, G4 and so many others... are changing versions at each release). Therefore an automatic cvsup procedure would help but only for the external libraries handled by ATLAS. Those requiring the action of system administrators cannot be dealt with automatically. However the adequate information is needed.

b.) Problems during the data processing:

No problems have been encountered during the data processing. We have been using the BQS batch queuing system.

c.) Problems getting data to/from CERN

No problems have been encountered during the data transfer To/From CERN. We have used bbftp through wacdr.cern.ch (CCin2p3/HPSS<->CERN/CASTOR).

9.5 Germany

a) Software installation

Since a while we had been carrying out local ATLAS software builds in Munich and Karlsruhe, using the cvsup tool to transfer the software to the respective sites and CMT to build them. The software worked well and showed in the preliminary random-number-comparison-tests that it performs s just as the RPM distributions from Italy. Those were not really an option for us since getting super-user rights at FZK, where 8 HEP experiments share a computing environment is not foreseen.

On a side note: Although everybody seemed to be happy with the RPMs, I do not understand why we had to use them: if the goal is to distribute the necessary executables and libraries only as opposed to set-up a full blown ATLAS software development environment, why not really ONLY distribute those needed files? Especially since CMT makes it easy to identify those components: you only need to look in the build directory, usually 'Linux-gcc-opt'. I tried it out and produced an 18 MB (uncompressed) directory, which contained ATLSIM, and every thing needed, and it passed the above-mentioned test as well.

b.) Problems during the data processing:

We have been running a little less than 3000 jobs a 200-500 events (~24-48 h) and only one single job terminated for unknown reasons. The rest of the 'problems' consists of partitions which crashed due to the

"CALOR: FATAL ERROR IN EVAP"

²⁴ <http://isnwww.in2p3.fr/atlas/fairouz/dc/dcl.html>

error, which occurred in about 100 of the 3000 jobs. Those jobs had to be rerun with a different random number seed and finished all successfully the second time.

c.) Problems getting data to/from CERN

After a long and difficult installation procedure of bbftp at our site, data transfer of the root input partitions has been working smoothly ever since.

9.6 Israel

a) Software installation

We had the Release 3.2.1 software already installed at our site. Nevertheless, we have installed the right RPM distribution also. But we had to have super-user privileges to install the RPMs, due to the fact that paths were absolute and not relative. Then, moving the software in the desired location required fixing the symbolic links, which unfortunately are again absolute and not relative as they should be.

b.) Problems during the data processing:

Only one ZEBRA file had to be rerun with another RANLUX parameter to avoid the CALOR STOP problem.

c.) Problems getting data to/from CERN

Downloading the root files was not easy, we had to do it in two steps: first, a rfcp from castor to /tmp on a machine at CERN, then a rsync from our site.

9.7 Italy

a) Software installation

All the sites have installed the software via the RPMs, without any big problem. In two sites (CNAF and Rome1) the installations have been also done using LCFG on EDG machines.

Just one comment for the RPMs: many people complained about the non-relocability of the packages. This issue has been now (probably) solved and the new relocatable RPMs are available at the same place where they were before (I just overwrote the old ones)²⁵.

This should also solve the problems when the user doesn't have the root password, since the whole kit may be relocated in a directory whose owner is not root. If somebody needs more instructions, please let me know and I'll post them in the list.

b.) Problems during the data processing:

No big problems. Single muon production has not showed any apparent problem, while for the rest only < 2% of the jobs have been resubmitted after a fatal stop (CALOR: Error in EVAP--> STOP). We have anyway to think about a new strategy for such kind of errors, since to change the RANLUX by hand for each job, when the error occurs, is probably not the best solution.

²⁵ <https://classis01.roma1.infn.it/ATLAS-farm/ATLAS-kit/3.2.1-2>.

c.) Problems getting data to/from CERN

We had two kinds of approaches:

- Roma1: data were copied on LXPLUS, via rfiio, and then transferred to the local farm via scp and vice versa;
- The Rest: data were copied directly to/from the local site via ftp using wacdr.

For the farm in Roma1 we had to execute that "double step" since the ftp protocol has been disabled in both directions for security reasons. There were no particular problems in transferring data to/from CERN, except that sometimes the transfer was really slow, but it does depend on the connection and was solved, in some cases, by transferring using multiple streams.

9.7 Japan

a) Software installation

We didn't see any problem in installing the software with the RPM kit. It was easy and straightforward because the farm is dedicated only to ATLAS and the root user was in our group and working closely. It would have been nicer if we could decide the directory to which the software is to be installed.

Although, there is a worry to have an RPM kit in addition to another installation. Before the RPM installation, we have 'copied' the ATLAS releases to our system. We didn't see any interference between these copied software and the RPM one this time, probably the creator of the kit must have been very careful, but it is worrying to have two trees of the same software, especially when the two can be different versions.

A good feature of the RPM kit was that it was 'frozen'. During the DC1-0, we had experienced that the software in the copied tree didn't produce the same results as the other sites. This was caused by changes in the release under /afs/cern.ch after we copied the files and also by building some binaries (libraries) locally. Building the libraries at CERN and then copying them to Tokyo solved the problem. (And also by copying the up-to-date source files once the difference was confirmed)

b.) Problems during the data processing:

No known big 'problems', but some comments. It was, however, worrisome and tiring that we had to make changes to the job scripts for site specific parameters every time we get a new script because we can overlook something and can make mistakes, and had to repeat the same things. It would be nice to have site-dependent part separately (even in the same script file).

No recipe to check production results was provided at first. One had to check the log files without knowing what to look for. The errors reported to the ML and what to do with them were accumulated quickly, and it was useful. But it would be nicer to have the information on the web page. It would also be nice to have a checklist and even a script to check log, data size, and so on.

The ones who execute the jobs not experts, and a few experts (or a single one) cannot look after all the productions.

Rerunning with a different random number seed added some complexity to the process. Although, this was understandable since the situation had not been foreseen.

We need to reconsider how to treat this for the future productions.

Some troubles occurred mainly due to the fact that our farm was really new and not yet well configured.

c.) Problems getting data to/from CERN

We transferred the data using rfc+scp, rfc between CASTOR and CERN machines and scp between CERN and Tokyo.

No problem was seen. Although the transfer was slow, I copied multiple files in parallel, and both downloading and uploading were finished in several hours.

One last comment:

It was a good exercise for our newly built pc-farm + batch system.

9.8 Nordu-Grid

Nordu-Grid has made a comparison of different processors. They found some differences between Pentium and Athlon processors (Athlon processors tend to be less efficient.)

a.) Software installation:

ATLAS software was rebuilt at most sites; hence the pseudorandom number sequences may diverge from the "standard" (a la CERN) ones.

b.) Problems during the data processing:

Problems: few jobs crashed (mostly "ZEBRA banks screwed up"), but were successfully re-run with the altered random seed (+1000000)

9.10 Russia

a.) Software installation:

We are using the "rsync" copy of the full /afs/cern.ch/ATLAS/software/dist/3.2.1 tree. It is linked to the faked local directory /afs/cern.ch/... The same is done for general CERN libraries and /afs/cern.ch/ATLAS/offline. The production scripts are CERN-oriented. It would be good to make them more flexible from the very beginning and single out site-independent core.

b.) Problems during the data processing:

Some lack of information at the first stage (hard to follow "official" line if you are not present at last DC meeting). Improved with new DC-page, but need some more efforts. In my opinion it would be nice to have a PBS version of production scripts (I think most of sites use this batch system)

The main problems were job interruptions in GCALOR

CALOR: Fatal ERROR in EVAP =====> STOP

requiring rerun with changed RUNLUX. This error is not detected automatically in the script; so, it requires manual job results monitoring. The solution is hardly acceptable for mass simulation and needs to be changed.

In addition we had some minor problems since several input EVGEN files were corrupted and they were regenerated. Several jobs were interrupted with diagnostics "segmentation fault" but they were caused by H/W problems at one of our CPUs.

c.) Problems getting data to/from CERN

Some weird problem with my 1701-1720 partitions on castor, since files were not overwritten by newer version; this was solved by deleting the old version and copying once again. Otherwise there were no real problems since the rather moderate volume of transferred data. We used ftp to/from wacdr.cern.ch.

9.11 Spain

a) Software installation

The RPM format was a great advance from previous distribution formats. It would be desirable to have it relocatable.

b.) Problems during the data processing:

1 of 400 jobs in partition 002000 gave CALOR: Error in EVAP--> STOP
9 of 1000 jobs in partition 002030 gave the same

c.) Problems getting data to/from CERN

We used ftp (wacdr.cern.ch) to get and to put the data files. The problem with ftp is that it is difficult to automatise.

9.12 Taiwan

a) Software installation

We had installed the ATLAS software by a complete mirroring of the ATLAS software directory under CERN AFS. The fake AFS directory was created locally although we do not have the AFS system in our PC farm. The program compiled and ran successfully at our site. However the test run using this produced a different random number sequence for the same set of events with the same random number seed. We then installed the RPM version, which produced exactly the same random number sequence for the test run.

There are pros and cons of the different software installations. To summarise, before we make the remote CVS/CMT checkout works, the RPM installation is still preferred since it guarantees the equality of the data produced at different sites under different machines and OS. The key point is to keep using the same compiler.

b.) Problems during the data processing:

There were a series of run-time errors during the production, which have been communicated among the DC people already. I shall not mention them here.

One type of "error" happened when a long-running job hung, mostly on the 500 MHz CPU, the job then hangs forever while producing a huge log file with repeating error messages.

This kind of errors often disappear if re-run on the faster CPUs. Probably it was due to the screwing-up in the memory.

c.) Problems getting data to/from CERN

Currently the network connection from Taiwan to CERN is via the Japan-US-Europe route. A traceroute command shows that it passes APAN net in Japan, and then the Abilene gateways in US, before it goes over the SWISSCOM switch.ch into CERN.

A single stream ftp session to CERN castor via disk server wacdr.cern.ch produced about 95 KBytes/s transfer speed only. We have tried different programs and different ways of data transfer, from the multi-session normal ftp to the multiple stream capable programs such as bbftp. It proved that the simultaneous multiple stream data transfer improved significantly the effective bandwidth for data transferring. We used normally 10 streams, which produced an effective average transfer speed of about 1 MBytes/s between our hepfarm and the CERN castor.

We look forward to using the GridFTP for data transferring, which is multiple-stream, enabled. For the near future the ASCC is investigating for a several Gbps direct connection to StarLight, which will dramatically enhance the effective data transfer bandwidth between Taiwan and CERN.

Problems happen from time to time with the remote access to the CERN castor storage via the wacdr.cern.ch disk server. Sometimes it was due to the system development being performed by the castor support team at CERN. For example, we had difficulties to get connection to wacdr.cern.ch few days ago, and were later told by the support team that they were testing using different ports and different ftp modes (either active mode and/or passive mode). In principle, we have succeeded for most of the time in transferring to/from the CERN castor.

9.13 UK : Liverpool MAP (Monte Carlo Array Processor)

We have a 300 PC array (worker nodes) each with 20G of available disk space. One of 6 storage nodes (compass nodes) broadcasts the required software and data to each of the worker nodes at the start of a job. The required output is then transferred back to the compass node upon the conclusion of the job and stored on the 500GB we have available per compass node.

This architecture thus differs significantly from a PBS batch system set up on a PC farm. We have therefore encountered some (possibly unique?) challenges that had to be worked around:

a) Software installation

There was no problem in installing the software on the compass node. However, as mentioned above the worker nodes do not retain any data after the conclusion of a job. As a last resort, it would have been possible to install the RPMs on each worker node. This would have been very time consuming for 300 PCs and is certainly not a scalable method.

A mount point to the compass node was also an option. This is fine for relatively small farms but was unworkable when 300 PC are trying to access the same file simultaneously.

The only workaround available was to broadcast the contents of /opt/ATLAS to each worker node at the start of each job. Given the size of the whole directory tree (>1G) this seemed quite inefficient. Indeed, the contents of /opt/ATLAS were stripped down to only what was essential (relating to only 10% of the total file size). Ideally a static executable would have been preferred (a la LHCb) in this instance.

b.) Problems during the data processing:

The second problem was the large size of the MC file (2G). This had to broadcast to each node in its entirety although only a fraction of the file would be processed. This again slowed down the start of each job. Could these files be split up prior to execution? Or could an individual institute generate the MC rather than retrieval from a central repository?

These comments (a.) and b.) have been from the experience gained from running over 500k events in dataset 002000 - so the next point is relevant for this dataset at least.

From the scripts, the default number of events for one job to run over is 5000. It seems that the naming system of the output files is dependent on this being fixed. Ideally, we would have liked to run on a smaller number of events. This would have given us the opportunity to spread 1 partition (100k events) across 300 nodes (or say, 500 events across 200 nodes) to enable a much more economical production and a full use of our resources.

In order for the 5000-event structure to be retained we had to resort to installing 5 separate queues. Thereby splitting MAP into 5 farms of 20 PC's each. This meant discarding ~60% of the total processing power (or ~200 PC's not used) we had available for MC generation.

One solution for a previous run of LHCb MC generation was to split the MC file up but then merge the ZEBRA output upon completion of the job. Would that be possible for ATLAS? That would certainly be an easy solution to our problem.

The large job time is also a factor since MAP is a shared resource with other experiments. A much shorter run time would be therefore more preferable for us.

c.) Problems getting data to/from CERN

No problems transferring data from CERN.

9.14 US Test-bed

a) Software installation

For dc1 grid production in the U.S., we used a tarball made by Pavel Nevski at BNL (containing binaries only for Red Hat Linux). The executables were installed on Globus gatekeeper machines at 3 (out of the 8) U.S. test-bed sites.

b.) Experience and problems during the data processing:

We used a grid scheduler to submit the jobs. This scheduler automatically submits about 60 jobs a day, wherever it finds available capacity among the 3 sites. A second scheduler was run independently to achieve a rate of ~120 submissions a day. We had a 80% success rate - most failures happened due to hardware and software failures or scheduled outages that we understand and should be able to improve next time. The submission process was completely automatic and required very little supervision or intervention (in most cases, if a site was unavailable, the scheduler continued production with the other available sites without problem).

Each production job on the grid had many stages. First the Globus gatekeeper of the site selected by the scheduler is queried for software location information. Next, a suitable available partition is chosen for production. The proposed LFN is registered in Magda along with various production related information. All executables are staged into a temporary location. A script with the location of the executables and environment variables is sent to the queue on the selected site. The job is started asynchronously by the batch queue system. The scheduler checks every 5 minutes if the production job has finished. Once it finishes (on average after 14 hours), the files are moved to the BNL HPSS storage system by Magda. All LFNs are registered in the Magda catalogue. A replica is also made by Magda at one of the available grid sites.

An independent semi-automatic quality of service (QOS) process is run periodically. This job checks the Magda production database for job status (the production job updates this database periodically during staging and execution) of every partition. It checks job status on the submitted queue through Globus. It verifies through Magda that all files are correctly stored in the HPSS and replica locations. It checks if the temporary staging location has been cleaned up after production. This process can correct for many failures and updates the production database if it can recover files. For example, the BNL HPSS was unavailable for a couple of days - production continued without any changes. When HPSS was available again, the QOS process automatically copied and catalogued all primary files from the replicas using Magda.

Most of the problems during the 2 weeks of production are typical of distributed systems spread out over 4 locations thousands of miles away (New York, California, Texas and Oklahoma). Various machines were not available at critical times. Even when empty queues were available, however we could not run production faster than about 70-80 jobs per day at any one site. We are taking steps to fix this limitation.

Magda has been used in DC1 in three aspects:

- **US grid test-bed production**
- **Automatic transferring files between BNL HPSS and CERN CASTOR.**
About 3 TB data has been copied using Magda. gridftp or bbftp were used for the trans-Atlantic transferring.
- **Cataloguing files by using the file spider.**
About 12K primary ZEBRA file instances are registered in Magda.

c.) Problems getting data to/from CERN

Not sent any data to CERN yet - all data is being stored at the Brookhaven tier 1 site.

9.15 USA : BNL

a) Software installation

BNL used its own installation of the production software, which encapsulated ALL components in one tarfile. All job control parameters are coming from a database (Virtual Data Catalogue). All jobs were running using a set of few common NFS discs.

b.) Problems during the data processing:

During the production the following problems where observed:

- 11 jobs have to be restarted with a different random seed due to GCALOR problem
- NFS server was taken for the maintenance twice during this period (July 11 - September 1st), thus leading to the loss of all running jobs twice.

No new problem was encountered during the production period.

c.) Problems getting data to/from CERN

For the data movement from BNL hpss to CERN castor, we do it by three steps: BNL hpss -> BNLdisk cache -> CERN disk cache -> CERN castor. Generally the movement runs smoothly.

For the transferring of BNL disk cache to CERN disk cache, we run 'globus-url-copy' client at LXPLUS (some time ago we ran 'bbftp' client there). We use /tmp on LXPLUS as cache, and clean it up after we are done. I installed 'globus-url-copy' client and 'bbftp' client in my area. To avoid typing password, I have a doe certificate to run 'globus-url-copy'. (ran bbftp client + ssh-agent some time ago).

For the transferring of CERN disk cache to CERN castor, run ftp to wacdr.cern.ch locally.