# Distributed data analysis over the grid and a study of the decay MSSM A/H $\rightarrow$ tau tau $\rightarrow$ two jets at CMS

Stuart L. Wakefield
Imperial College London
2007

A Thesis submitted for the degree of
Doctor of Philosophy at the University of London
and for the
Diploma of Imperial College

## Declaration

Throughout this thesis where the work of others is presented references to the originating work are shown. Where my own work has been published references are also listed for completeness. The areas of this thesis that are my own work are given below.

I was responsible for the input data handling (Section 4.5.2), submitted workflow (Section 4.5.5) and local farm submission (4.5.7) parts of the distributed analysis package (GROSS [1]). Shortly after the first public release the main developer left leaving the author to take up this role. The project was then developed further, in response to feedback from users. All further work (Sections 4.6 onwards), including major structural and user interface changes necessary due to the new PubDB system, were implemented by me except for the low level http PubDB interaction which was written by a new developer under my guidance.

When development was moved to the BOSS [2] project, I was heavily involved in the design and architecture decisions (Section 5.3). The task concept and the use of XML for the task description was my idea. I was then responsible for writing the XML processing implementation (Section 5.4) as well as the multi-language API (Section 5.5).

In the $\tau$ study, Chapter 7, I determined the optimum reconstruction cone (Section 7.1), the simulated $\tau$ energy scale and a Monte Carlo calibration for the energy scale (Section 7.3.1). I also evaluated the effectiveness of a proposed method for obtaining the calibration from real data (Section 7.3.2).

The Higgs study (Section 8) involved a comparison with a previous study using a different experimental signature. As such this study is all my own work which further develops previous studies.

## Abstract

CMS is one of two general purpose detectors at the Large Hadron Collider at CERN. During each year of operation CMS will collect 10 petabytes of data, which must be reliably stored and made available for analysis by physicists around the world. CMS has adopted a computing model based on Grid technology. A prototype distributed analysis framework for CMS is presented. This framework enabled CMS physics analysis to be performed at distributed sites. An evaluation of this tool was performed by members of the CMS Higgs analysis group. The results of this evaluation were used in the redesign of the CMS distributed analysis software.

$\tau$ leptons can be used as a tool in the identification of several physics processes. In 65% of cases the $\tau$ decays hadronically to form a jet. The development of a Monte Carlo jet energy correction is presented along with a proposed method for obtaining the correction from real data using jet plus $\gamma$ events.

In the Minimally Superymmetric Standard Model (MSSM) heavy neutral Higgs boson decays copiously to 2 $\tau$ leptons. A study of $gg \to A/H \to 2\tau$ jets using missing energy for event selection is presented. The expected $\tan\beta$ sensitivity of CMS with 60 fb$^{-1}$ of data is presented and compared with previous studies which made use of b-tagging in the associated channel $gg \to bbH \to 2\tau$ jets.

**Acknowledgements**

I would like to thank Professor Dornan and PPARC for giving me the opportunity to study particle physics at Imperial College.

I would like to thank my supervisor, Dave Colling, for his guidance during the last three years and for teaching me HEP computing, which he very nearly made interesting. My one C++ course has been bolstered by Hugh Tallini and Kostas Georgiou. Hugh wrote the original version of GROSS and Kostas provided much-appreciated advice about the many changes applied later, as well as computing in general.

Sasha Nikitenko has been invaluable for guiding my physics analysis, especially given how little physics I knew when I started.

Given the nature of this thesis there are many people within the LCG/EGEE and CMS collaborations without whom this work would not have been possible. Thank you for your efforts.

If this thesis presents my work clearly and concisely it is only because of proof-reading by Dave Colling, Barry MacEvoy and Sebastien Greder.

I would also like to thank all those at Imperial who have made me glad I did not resign on my first day. I can't name them all, suffice to say it includes everyone that has bought me a drink after work.

Finally, I would like to thank my friends and family, especially my Mum without whom I would not be where I am now.

## Preface

The Standard Model has been tested to great precision and for the most part agrees well with experiment. To provide particles with mass it relies on the Higgs mechanism [3]. This introduces a new particle, the Higgs boson. However, there are features of the Higgs mechanism which perturb some physicists and so a complementary theory, Supersymmetry [4], has been proposed. This theory requires additional Higgs bosons. The nature of both these theories and experimental results suggest that the Higgs boson(s) have masses below 1 TeV/c$^2$.

The Large Hadron Collider [5] (LHC) currently being built at CERN has been designed to test these theories. The Compact Muon Solenoid [6, 7] (CMS) detector is located at the LHC and should either discover, or rule out, both the Higgs mechanism and low energy Supersymmetry. A description of the design and performance of this detector, concentrating on systems relevant to the analysis presented later, is given in Chapter 1.

The processes that will identify new physics are rare and will suffer from large backgrounds of well-understood physics. In order to obtain sufficient signal events the LHC must operate at unprecedented energy, luminosity and event rate. CMS will produce data at a rate of 1–10 PB a year for many years. This requires a new paradigm in data storage and analysis, which is provided by a worldwide grid of computational and storage resources made available to any physicist for their research. The grid technology and software designed for the LHC is described in Chapter 2.

The way in which CMS will use the grid is described in Chapter 3. CMS required an application to allow its data analysis software to take advantage of distributed resources. A prototype distributed analysis tool, called GROSS, was developed and is described in Chapter 4.

Many of the features demonstrated in GROSS were implemented in another CMS software system called BOSS. This process is described in Chapter 5.

Chapter 6 describes the Higgs mechanism, its problems and how supersymmetry may solve them. There are many flavours of supersymmetry, the simplest of which is known as the Minimal Supersymetric Standard Model (MSSM). The description of the MSSM in this chapter concentrates on areas useful for a search for heavy neutral MSSM Higgs bosons, the $A$ and $H$. It is shown why the $\tau$ lepton will provide a useful signature of these particles. Chapter 7 describes properties of the $\tau$, how to identify them at CMS and how to measure their energies correctly.

A study of CMS' ability to find these particles is presented in Chapter 8. This study investigates the possibility of seeing the process $gg \to A/H \to \tau\tau \to 2$ jets at CMS. It builds on a previous study which investigated $gg \to bbA/H \to \tau\tau \to 2$ jets with b-tagging used for event selection. The study presented here investigated the feasibility of replacing the b-tagging selection with one based on missing energy. This has the advantage of including the gluon fusion, $gg \to A/H$, production mechanism. The MSSM parameter space that CMS will be able to search in the early years of data-taking with this strategy is also presented.

This layout has been chosen to provide a more flowing thesis. The theory chapter was placed just before the $\tau$ study and analysis chapters so that it was close to the place where the information contained within was put to use.

# Contents

**1 The Large Hadron Collider and the CMS detector 10**

1.1 The Large Hadron Collider . . . 10

1.2 The CMS detector . . . 11

1.2.1 CMS coordinate system . . . 12

1.2.2 The Silicon Tracker . . . 12

1.2.3 The Electromagnetic Calorimeter . . . 15

1.2.4 The Hadronic Calorimeter . . . 16

1.2.5 The Muon Detectors . . . 19

1.2.6 The Trigger System . . . 21

**2 The Grid and the LCG 23**

2.1 Introduction . . . 23

2.2 The LCG . . . 23

2.2.1 Data Management . . . 25

2.2.2 Workload Management . . . 26

2.2.3 LCG object persistency . . . 26

2.3 Summary . . . 27

**3 CMS Computing Model 28**

3.1 Introduction . . . 28

3.2 Data Formats . . . 29

3.3 Tier roles & responsibilities . . . 30

3.3.1 Tier-0 . . . 30

3.3.2 Tier-1 . . . 30

3.3.3 Tier-2 . . . 31

3.3.4 Tier-3 . . . 31

3.3.5 CMS-CAF . . . 31

3.4 Data Distribution . . . 32

3.5 Computing Services . . . 32

3.6 Data Management System . . . 34

3.7 CMS Workflow Management System . . . 35

3.7.1 Introduction . . . 35

3.7.2 User analysis . . . 36

3.8 Alternative computing models within HEP . . . 38

3.8.1 Tevatron Experiments . . . 39

3.8.2 LHC experiments . . . 39
3.9 Summary . . . 41

**4 Distributed analysis with GROSS 42**
4.1 Introduction . . . 42
4.2 CMS Data Challenge 2004 . . . 42
4.2.1 CMS analysis framework . . . 43
4.2.2 Event data model . . . 43
4.2.3 CMS software available at sites . . . 43
4.2.4 Data replication and discovery . . . 44
4.2.5 BOSS . . . 44
4.3 The GROSS design Process . . . 46
4.4 Architecture . . . 49
4.5 Functionality and interface . . . 52
4.5.1 Task Creation . . . 52
4.5.2 Input data handling . . . 52
4.5.3 Task splitting . . . 53
4.5.4 Task submission . . . 55
4.5.5 ORCA wrapper script . . . 55
4.5.6 Querying and output retrieval . . . 57
4.5.7 Local farm submission . . . 58
4.6 Performance and useability . . . 58
4.7 Post DC04 distributed analysis . . . 60
4.8 CRAB . . . 60
4.9 New architecture and features . . . 61
4.10 Community Evaluation . . . 64
4.11 Conclusion . . . 64

**5 Boss development 66**
5.1 Introduction . . . 66
5.2 Limitations of previous versions . . . 66
5.3 New Architecture . . . 67
5.4 Task Definitions . . . 69
5.5 API . . . 71
5.6 Conclusion . . . 72

**6 Higgs mechanism and Supersymmetry 73**
6.1 Introduction . . . 73
6.2 Higgs mechanism . . . 74
6.3 Supersymmetry . . . 76
6.4 The Minimal Supersymetric Standard Model . . . 77
6.5 Summary . . . 79

**7 $\tau$ tagging and calibration** **80**
7.1 Introduction . . . 80
7.2 $\tau$ tagging . . . 80
7.2.1 Tracker isolation . . . 80
7.2.2 Electromagnetic calorimeter isolation . . . 82
7.2.3 Electron rejection . . . 83
7.2.4 High Level Trigger . . . 83
7.3 $\tau$ jet energy scale . . . 85
7.3.1 Monte Carlo calibration . . . 85
7.3.2 Proposal for calibration from data . . . 86
7.4 Conclusion . . . 88

**8 Study of A/H$\rightarrow \tau\tau \rightarrow$two jets** **90**
8.1 Introduction . . . 90
8.2 Signal and background generation . . . 92
8.3 Level-1 and high level triggers . . . 93
8.4 Offline reconstruction and selections . . . 93
8.4.1 $\tau$ Identification and reconstruction . . . 94
8.4.2 Missing energy . . . 97
8.4.3 Higgs mass reconstruction . . . 99
8.5 Cut overview . . . 99
8.6 Efficiencies and number of events . . . 101
8.7 Evaluation of background from data and systematics . . . 103
8.7.1 QCD multi-jets . . . 103
8.7.2 Z($\gamma$) $\rightarrow \tau\tau$ . . . 103
8.7.3 Total uncertainty . . . 112
8.8 Discovery reach in $\tan\beta - m_A$ plane . . . 113
8.9 Conclusion . . . 114

**9 Conclusion** **115**

**A Acronyms** **116**

**References** **119**

# Chapter 1

# The Large Hadron Collider and the CMS detector

## 1.1 The Large Hadron Collider

The Large Hadron Collider (LHC) [5] is a proton-proton collider currently being constructed at CERN, Geneva. It has been designed to achieve a centre-of-mass ($\sqrt{s}$) energy of 14 TeV and an instantaneous luminosity of $\mathcal{L} = 10^{34}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. As well as colliding proton beams the LHC will also provide Heavy Ion (HI) collisions at a centre-of-mass energy of 1,312 TeV and a luminosity of $\mathcal{L} = 10^{27}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$.

Proton bunches are formed in the 26 GeV Proton Synchrotron (PS) with 25 ns spacing. The beam will then be accelerated to 450 GeV by the Super Proton Synchrotron (SPS) [8] and transferred to the LHC ring. This process will be repeated 24 times, resulting in 2 counter-rotating beams each consisting of 2,808 bunches with a 25 ns spacing and containing $1.15 \times 10^{11}$ protons. Once circulating in the LHC, the bunches will be accelerated by 0.5 MeV/orbit by 1,232 r.f cavities until reaching 7 TeV.

The LHC will operate in a number of reduced modes over several years before reaching its design parameters. Initial commissioning will take place in the last few months of 2007 with beams consisting of just a few bunches. During this phase the r.f cavities will not be operational, limiting the beam energy to 450 GeV from the SPS.

Following this the machine will be commissioned with 7 TeV beams. This will take approximately 5 months and will leave the 2nd half of 2008 for a pilot physics run. This pilot run will have a 75 ns+ bunch spacing and a luminosity of $\mathcal{L} = 1 \times 10^{29} - 2 \times 10^{31}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ with the aim of taking 2–3 $\mathrm{pb}^{-1}$ of data.

Following the pilot run the first physics run will start in 2009 with a 75 ns bunch spacing. An instantaneous luminosity of $\mathcal{L} = 2 \times 10^{33}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ will be reached with an average of ∼5 inelastic collisions per bunch crossing. It is hoped that up to $5\mathrm{fb}^{-1}$ of data can be collected during this period.

The luminosity will be limited to $\mathcal{L} = 2 \times 10^{33}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ until at least 2010 while the beam dump and collimation systems are staged. During this time the bunch spacing will be reduced from 75 ns to 25 ns with the aim of collecting 10–30 $\mathrm{fb}^{-1}$ of data in the low luminosity régime. After this the luminosity will increase and finally reach the design value, $\mathcal{L} = 10^{34}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$, with 20 inelastic collisions each per bunch crossing.

## 1.2 The CMS detector

The Compact Muon Solenoid (CMS) [6, 7] experiment is one of two general purpose detectors that will operate at the LHC. It has been designed to detect the widest range of new physics possible [9, 10]. To facilitate this the main design goals were:

- Good muon identification;
- Good charged particle tracking;
- Good electromagnetic energy resolution; and
- Good missing transverse energy ($\mathrm{E}_\mathrm{T}^\mathrm{miss}$) and jet resolution.

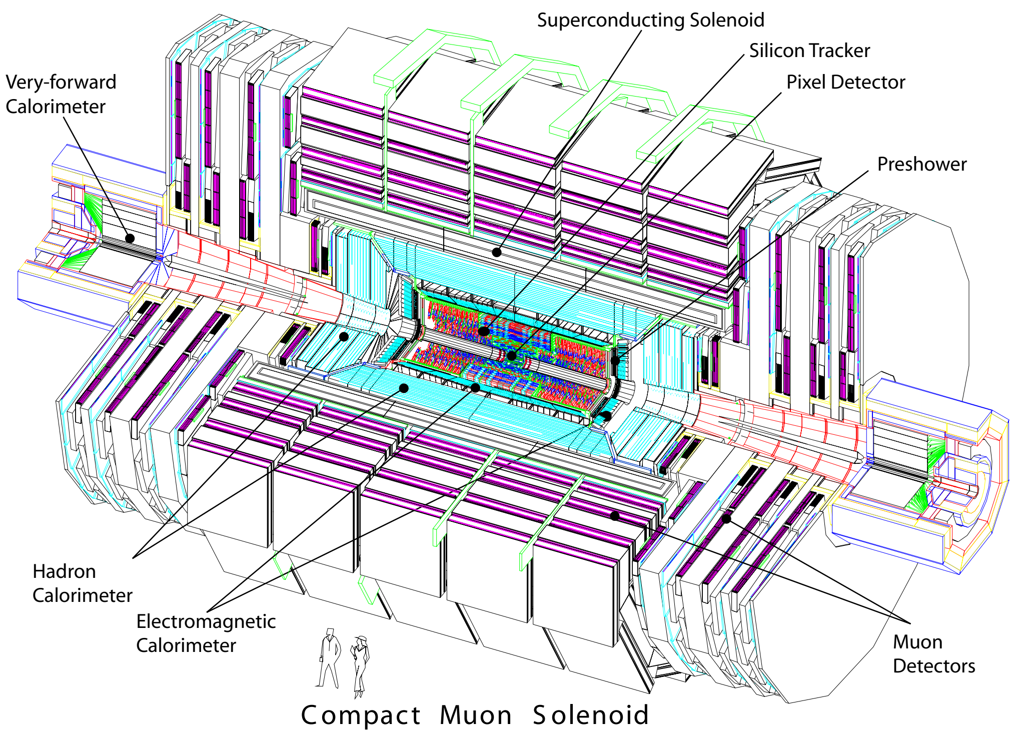


Figure 1.1: An exploded view of the CMS detector showing the major components. [9]

CMS, shown in Figure 1.1, is a large and complex detector composed of multiple subsystems and detectors. Major sub-detectors include a silicon tracking detector [11], a crystal electromagnetic calorimeter [12], scintillating hadronic calorimeter [13] and several muon detectors [14]. The CMS detector is 21.6 m long, has a diameter of 14.6 m, weighs 12,500 metric tons and has $\sim 10^8$ electronic readout channels.

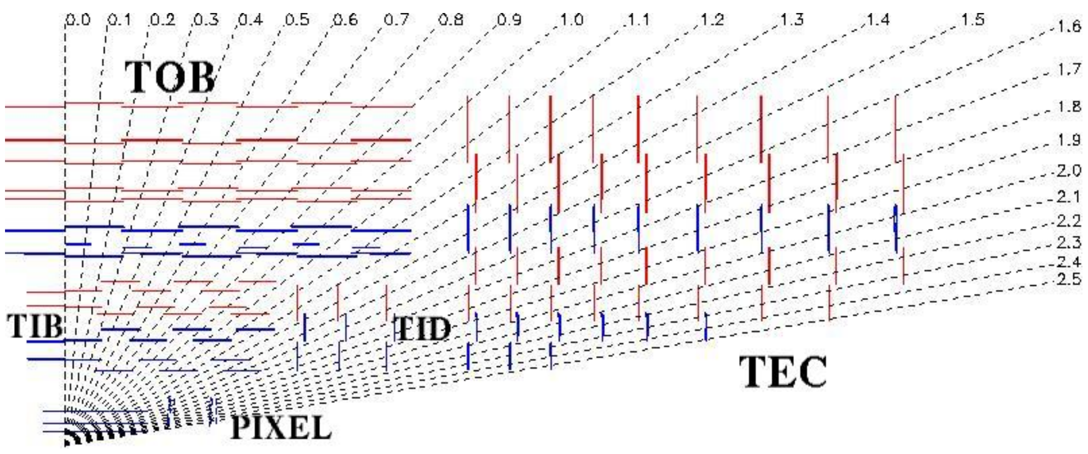


Figure 1.2: A one-quarter view of the tracker, showing all silicon detector layers. See the text for a description of the acronyms. Red (blue) layers use single (double) sided modules with dashed lines showing the pseudorapidity. From [9].

Charged particle momentum measurement is provided by the bending power of a 4T magnetic field. Provided by a 13 m long, 5.9 m diameter superconducting solenoid coil [15]. The coil current is 19.5 kA giving a total stored energy of 2.7 GJ. The magnetic flux is returned by a 1.8 m thick iron return yoke. The tracking and calorimetry systems are enclosed within the coil and the return yoke is instrumented with the muon detectors.

### 1.2.1 CMS coordinate system

In the CMS coordinate system the origin is centred on the nominal interaction point with the x-axis pointing radially inward towards the centre of the LHC ring, the y-axis pointing vertically upward and the z-axis pointing along the beam direction towards the Jura mountains. The azimuthal angle, $\phi$, is measured from the x-axis in the x-y plane. The polar angle, $\theta$, is measured between the line connecting the coordinate to the interaction point and the z-axis. Pseudorapidity, $\eta$, is defined as $\eta = -\ln\tan(\theta/2)$. Distance in the $\phi - \eta$ plane is measured as $\Delta\mathrm{R} = \sqrt{\Delta\eta^2 + \Delta\phi^2}$

Momenta and energy measured transverse to the beam, $\mathrm{p_T}$ and $\mathrm{E_T}$ respectively, are computed from their x and y components (i.e. $\mathrm{E_T} = \mathrm{E_x} + \mathrm{E_y}$). The energy imbalance measured in the transverse plane is denoted by $\mathrm{E_T^{miss}}$, where $\mathrm{E_T^{miss}} = -\mathrm{E_T}$.

### 1.2.2 The Silicon Tracker

The closest sub detector to the interaction point is the silicon tracking system. It measures the trajectories and momenta of charged particles up to $|\eta| \lesssim 2.4$. CMS has opted for a full silicon tracker providing a relatively low number of precise measurement points instead of a continuous tracking technology. Two different silicon technologies are used: pixel and microstrip detectors. Close to the beam pipe the high particle flux requires small pitch pixel detectors whereas further out the occupancy drops sufficiently to allow silicon microstrip detectors to be used. The tracker, shown in Figure 1.2, has an outer radius of

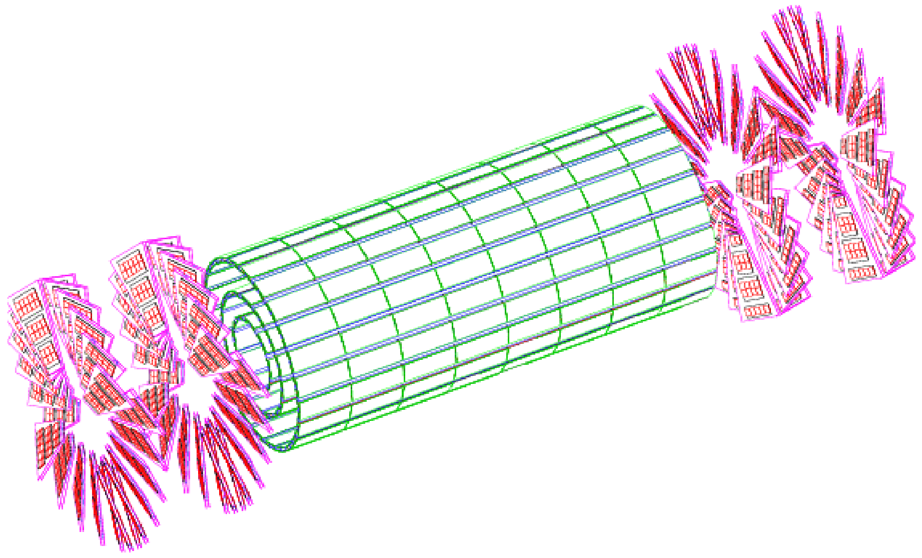


Figure 1.3: Layout of pixel detectors in the CMS tracker. [9]

110 cm, is 540 cm long and has $\sim 44 \times 10^6$ readout channels.

The pixel detector, shown in Figure 1.3, consists of 3 barrel layers with 2 pairs of endcap disks and provides two hit coverage up to $|\eta| = 2.2$. The barrel layers are located at radii of 4.4 cm, 7.3 cm and 10.2 cm and are 53 cm long. The two endcap disks are located at $|z| = 34.5$ cm and 46.5 cm with radii of 6 cm to 15 cm.

In order to maximise the vertex resolution the pixel pitch is $\approx 100 \times 150$ mm$^2$. The pixel spatial resolution is increased by using analogue signal interpolation of the charge sharing induced by the large Lorentz drift in the magnetic field. Thus the barrel pixel layers are collinear to the beam and the endcaps are arranged in a turbine-like geometry with blades rotated by 20°. The spatial resolution is 10 $\mu$m in r-$\phi$ and 20 $\mu$m in z giving a vertex resolution of $\sim 40\,\mu$m.

The silicon strip tracker surrounds the pixel detector and covers the range $|\eta| < 2.4$. The strip tracker is split into 2 systems, inner and outer, each of which has barrel and endcap sections. Most detector layers utilise single sided microstrips but some use "stereo" modules consisting of two tilted back-to-back modules which provide a 3D hit measurement. The modules used in the various parts of the silicon tracker are listed in Table 1.1.

| part | No. detectors | thickness ($\mu$m) | mean pitch ($\mu$m) |
|---|---:|---:|---:|
| TIB | 2724 | 320 | 81/118 |
| TOB | 5208 | 500 | 81/183 |
| TID | 816 | 320 | 97/128/143 |
| TEC | 2512 | 320 | 96/126/128/143 |
| TEC(2) | 3888 | 500 | 143/158/183 |

Table 1.1: Detector types in the silicon tracker. See text for an explanation of the acronyms. [9]

The Tracker Inner Barrel (TIB) is composed of 4 microstrip layers covering $|z| < 65$ cm, where the first 2 layers use stereo modules with a stereo angle of 100 mrad. The silicon sensors have a thickness of 320 $\mu$m and a length of 10 cm. Point resolutions of 23–34 $\mu$m in r-$\phi$ and 230 $\mu$m in z are obtained depending on the layer. Each inner endcap, Tracker Inner Disk (TID), has three microstrip layers, the first two of which have stereo modules. The microstip detectors are 320 $\mu$m thick with a minimum length of 10 cm. A point resolution of 23–34 $\mu$m in r-$\phi$ and 230 $\mu$m in z is obtained.

The outer barrel system, Tracker Outer Barrel (TOB), comprises 6 layers covering the range $|z| < 110$ cm with the first two layers using stereo modules. The point resolution varies from 35–52 $\mu$m in r-$\phi$ and is constant at 530 $\mu$m in z. The Tracker EndCaps (TEC) comprise 9 layers over the range $120 < |z| < 280$ cm. The first two and the fifth disks have stereo modules. The microstrips have a thickness of 320–500 $\mu$m and a length of 25 cm. A point resolution of 35–52 $\mu$m in r-$\phi$ and 530 $\mu$m in z is obtained.

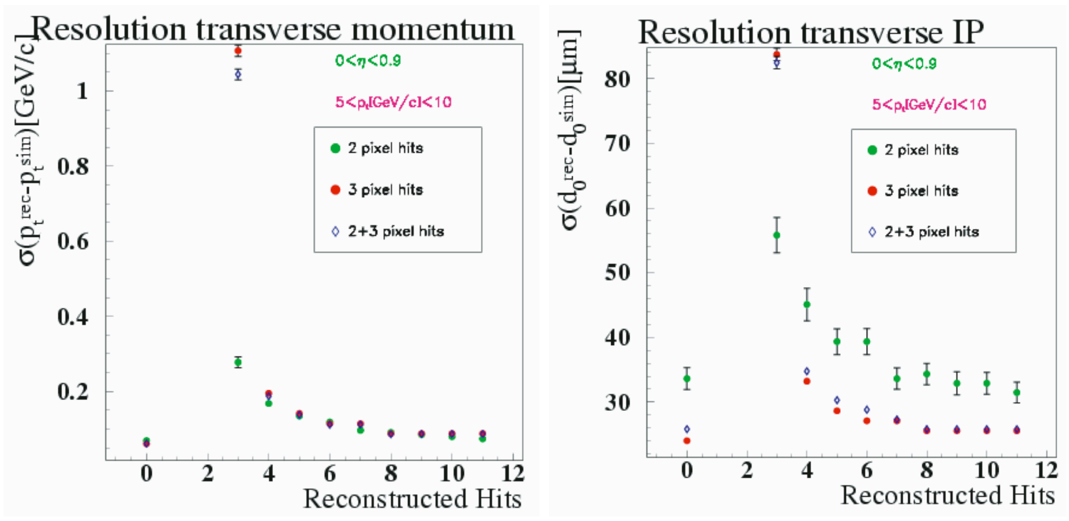


Figure 1.4: The tracker $p_T$ (left) and transverse impact parameter, d0, (right) resolution as a function of the number of reconstructed tracker hits.

Figure 1.4 shows the tracker performance as a function of the number of reconstructed hits. It can be seen that performance with only the pixel layers operational is reduced. Operating in this mode is significantly quicker than using the full tracker due to the reduced number of read out channels and combinatorials and is used for triggering when less precise but fast measurements are required.

Figure 1.5 shows the tracker $p_T$ resolution and global tracking efficiency for muons with $p_T$ of 1, 10 and 100 GeV/c. Global efficiency is defined as the reconstruction efficiency for all tracks, taking into account tracker acceptance, hit efficiency and pattern recognition efficiency. The $p_T$ resolution can be parameterised:

$$\frac{\sigma_{p_T}}{p_T} = a p_T \oplus 0.5\% \tag{1.1}$$

with $p_T$ in TeV and a = 15 for $|\eta| < 1.6$ and 60 for $1.6 < |\eta| < 2.5$ [9].

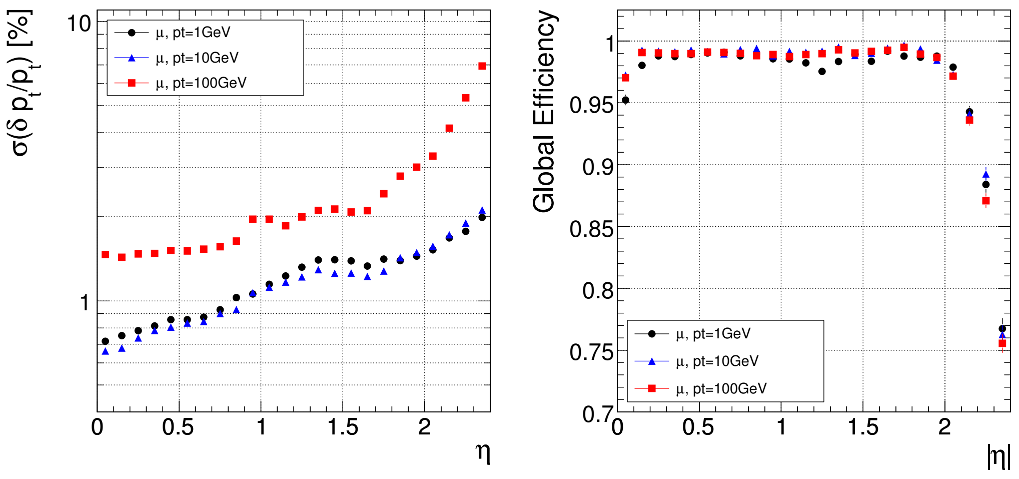


Figure 1.5: Tracker performance for muons with transverse momenta of 1, 10 and 100 GeV/c: transverse momentum resolution (left) and global track reconstruction efficiency (right). [9]

### 1.2.3 The Electromagnetic Calorimeter

Surrounding the silicon tracker is the electromagnetic calorimeter (ECAL). The ECAL is divided into a barrel and 2 endcap sections, shown in Figure 1.6. CMS has chosen scintillating lead tungstate ($PbWO_4$) crystal to provide precise electron and photon energy measurement. Lead tungstate crystals are radiation hard, have short radiation lengths ($X_0 = 0.89$ cm) and are fast (80% of light is emitted within 25 ns). This choice allowed a fast, fine grained and compact ECAL which could be placed inside the coil. $PbWO_4$ crystals yield a relatively low number of photons ($30\gamma/$ MeV) and so need to be read out by photodetectors with an intrinsic gain as photo multipliers cannot operate in the high magnetic field.

The ECAL barrel (EB) begins at a radius of 129 cm and covers the range $|\eta| < 1.479$. 61,200 crystals are grouped together to form one of 36 identical "supermodules". Each supermodule covers half of the barrel length. The crystals are mounted 3° off axis from the nominal vertex position to avoid energy leakage between crystals. Each crystal measures $22 \times 22 \times 230\,\text{mm}^3$ (25.8 $X_0$) and covers an area of $\Delta\eta \times \Delta\phi = 0.0174 \times 0.0174$. These crystals are read out by silicon avalanche photodiodes (APDs) with a gain of 50.

The ECAL endcaps (EE) are located 314 cm from the vertex and cover the range $1.479 < |\eta| < 3.0$. Each endcap is constructed from two "Dees" consisting of semi-circular aluminium plates mounting crystals in groups of $5 \times 5$ crystals, known as "supercrystals". The endcaps crystals are tilted off axis in an x-y grid. A total of 21,528 crystals of dimensions $28.6 \times 28.6 \times 220$ mm$^3$ (24.7 $X_0$) are used in the endcaps. Vacuum phototriodes are used for the endcap readout as they are more radiation hard than APDs.

A preshower detector comprised of two planes of lead absorber followed by silicon strip detectors is placed in front of the endcaps, covering $1.48 < |\eta| < 2.6$. These are used to help identify neutral pions in the endcap region where the average pion energy is high

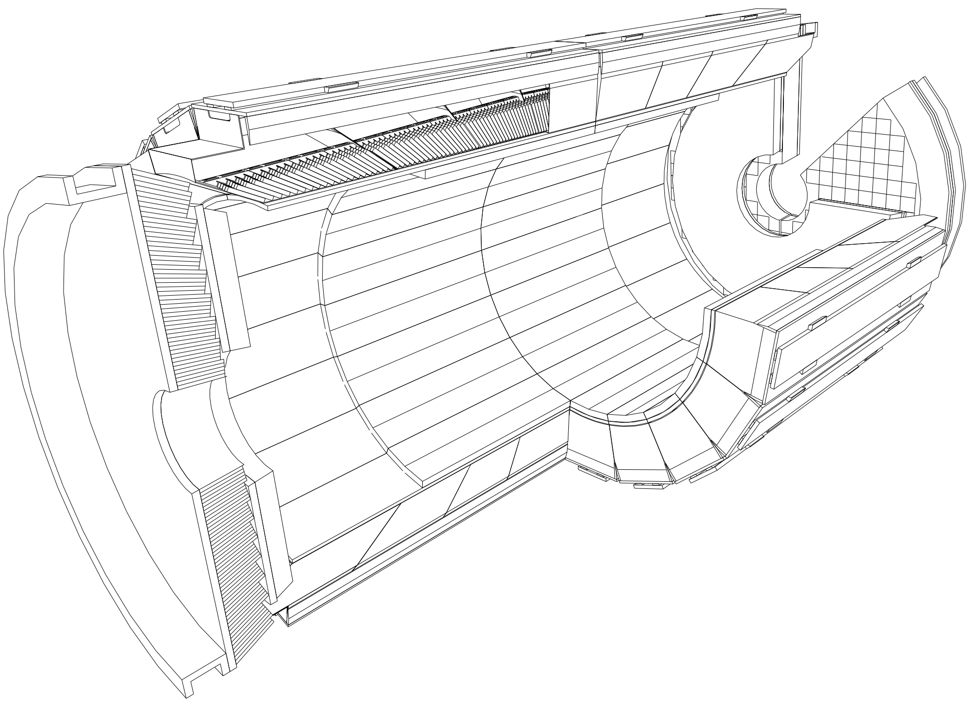


Figure 1.6: A 3D view of the CMS electromagnetic calorimeter. [12]

enough to make resolving individual photons difficult given the calorimeter granularity.

The ECAL performance has been evaluated in a test beam [9]. High energy electrons ($20 \leq \mathrm{E_T} \leq 250\,\mathrm{GeV}$) were used with a full barrel supermodule. The measured electron energy resolution is shown in Figure 1.7. The resolution can be parameterised as

$$\frac{\sigma}{E} = \frac{3.63\%}{\sqrt{E}} \oplus \frac{0.124\%}{E} \oplus 0.26\%, \tag{1.2}$$

where E is the beam energy in GeV, the first term is the stochastic term, the second the noise and the third the constant term. The Stochastic term comes from fluctuations in lateral containment and photostatistics [9], the noise term originates from preamplifier, digitisation and pileup noise, and the constant term comes from energy leakage and inter-crystal calibration.

### 1.2.4 The Hadronic Calorimeter

Surrounding the electromagnetic calorimeter is the hadronic calorimeter (HCAL). In order to minimise non-Gaussian energy resolution tails and to provide good containment the HCAL was placed inside the magnet coil. This required a compact absorber and left little room for the active medium. Brass was chosen for the absorber because it is non-magnetic and has a short interaction length ($\lambda$). Tile/fibre technology was chosen for the active medium together with wavelength-shifting fibre readouts.

The barrel section, denoted HB, spans the region $|\eta| < 1.4$ and is read out in towers of $\Delta\eta \times \Delta\phi = 0.087 \times 0.087$ in a single longitudinal sampling. The HB has 15 brass plates comprising $6.5\,\lambda$ thus additional layers, known as the hadron outer (HO), are placed behind

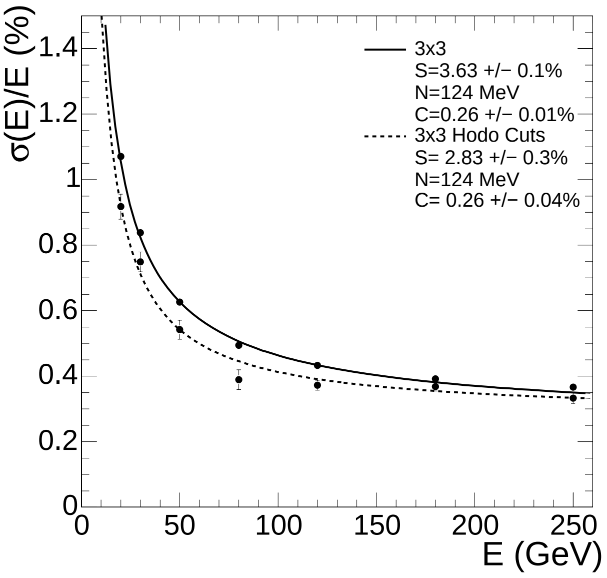


Figure 1.7: ECAL electron test beam data energy resolution as measured by an array of 3×3 crystals. The upper (lower) series shows events taken with a 20×20 (4×4) mm$^2$ trigger. The fit parameters correspond to those in Equation 1.2. [9]

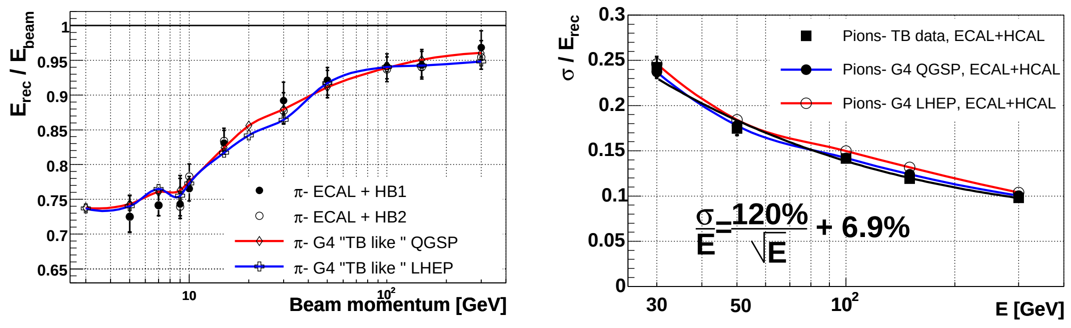


Figure 1.8: Nominal raw energy response and fractional energy resolution as a function of ECAL+HB energy for $\pi^-$. The fit for the non-compensated energy resolution is shown. Results for both testbeam and simulated (labeled G4) data are shown. [9]

the coil, hadron outer (HO), to act as a tail catcher. This covers the region $|\eta| < 1.26$ and uses the same tower geometry as the barrel. The HO extends the barrel HCAL depth to over 10 $\lambda$.

Hadron endcap (HE) disks are located either side of the solenoid coil and span the region $1.3 < |\eta| < 3.0$ with towers varying in size from $\Delta\eta \times \Delta\phi = 0.087 \times 0.8 - 0.35 \times 0.8$.

To extend the $\eta$ coverage, a forward calorimeter, hadron forward (HF), is located at the edges of CMS covering the region $3.0 < |\eta| < 5.0$ (see Figure 1.1). This detector is constructed from steel and quartz fibre. These materials lead to shorter and narrower jets which is useful in the high flux forward region.

Test beams have been used to evaluate the HCAL performance and characteristics. The 2004 HCAL test beam used a replica of a slice through the CMS detector with a $\pi^-$ beam [9]. The prototype detector included an aluminium slab, representing the solenoid,

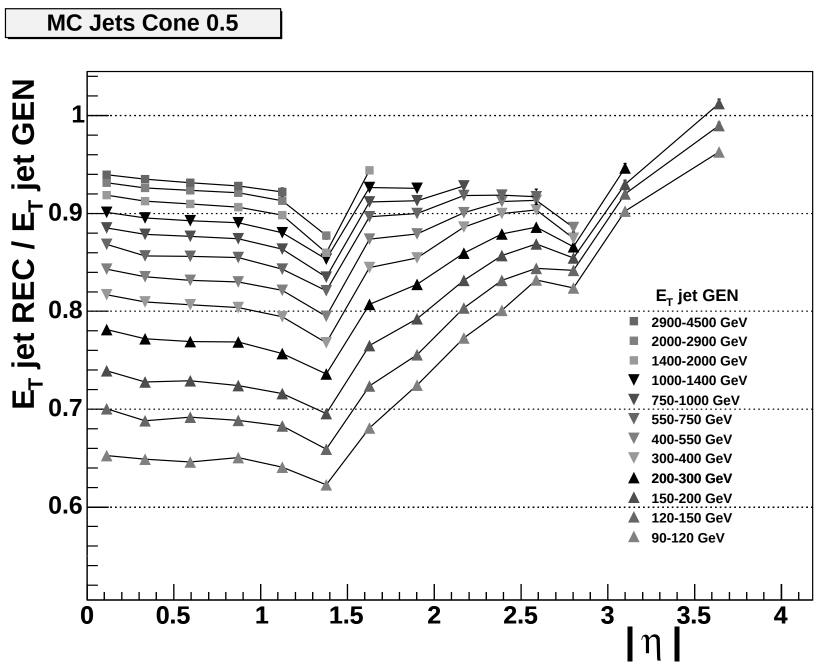


Figure 1.9: The ratio of the reconstructed jet transverse energy $E_T^{rec}$ to the generated transverse energy $E_T^{MC}$ as a function of generated jet $\eta$ for jets with different $E_T^{MC}$ reconstructed by the iterative cone $R = 0.5$ algorithm. [9]

a prototype ECAL detector and 144 HB and 60 HO towers. This detector was exposed to $\pi^-$ beams of energies 2-9 GeV and 10-300 GeV. The single particle energy response and resolution are shown in Figure 1.8 both for test beam and simulated data. The energy resolution can be parameterized as:

$$\frac{\sigma}{E} = \frac{120\%}{\sqrt{E}} \oplus 6.9\% \tag{1.3}$$

The HCAL jet and $E_T^{miss}$ performance have been evaluated with simulated QCD multi-jet events at $\mathcal{L} = 2 \times 10^{33}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ [9]. The jet energy response for jets of various generated $E_T$ ($E_T^{MC}$) is shown in Figure 1.9.

The granularity of the different HCAL detectors has been chosen to provide similar jet energy resolutions for all regions. The barrel jet energy resolution can be seen in Figure 1.10 and can be parameterised:

$$\frac{\sigma\left(\frac{E_T^{rec}}{E_T^{MC}}\right)}{\langle\frac{E_T^{rec}}{E_T^{MC}}\rangle} = \frac{5.6}{E_T^{MC}} \oplus \frac{1.25}{E_T^{MC}} \oplus 0.033 \tag{1.4}$$

where the first term is the noise term, the second the stochastic term and the third the constant term. The noise is due to fixed energy fluctuations from electronic noise, pileup and the underlying event, the second term is the stochastic calorimeter response and the third is a constant term originating from the non-uniformities and non-linearities of the calorimeter response.

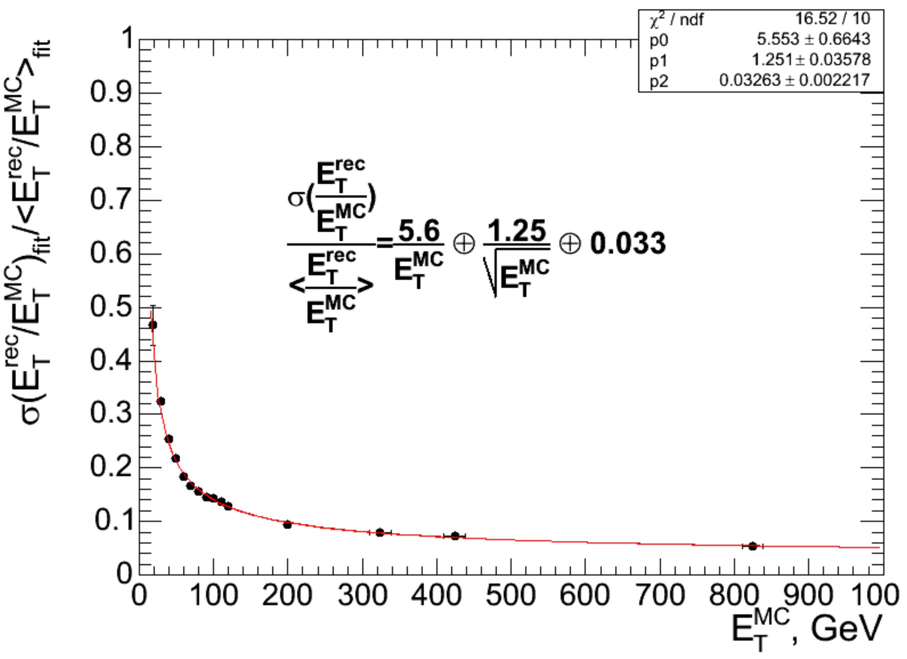


Figure 1.10: The jet transverse energy resolution for simulated jet events with pileup in the HCAL barrel. [9]

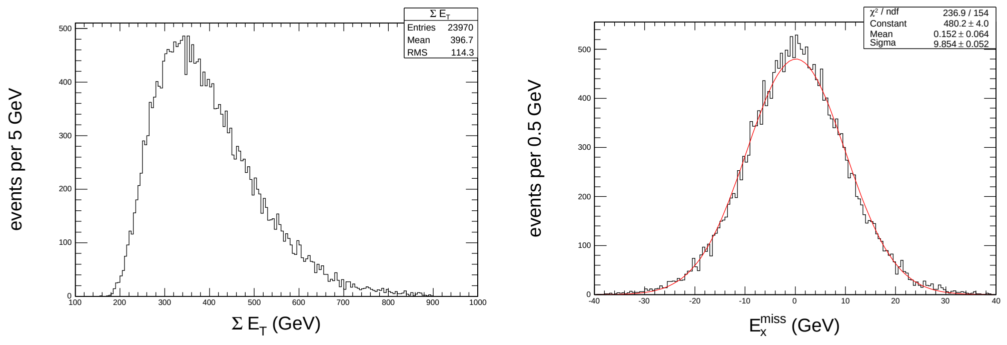


Figure 1.11: Distribution of sum $E_T$ (left) and $E_x^{miss}$ (right) for soft QCD events with pileup. The $E_T^{miss}$ resolution of 9.9GeV is in good agreement with Equation 1.6. [9]

The missing energy $E_T^{miss}$ mean and resolution distributions for soft ($0 < p_T < 15\,\text{GeV}$) QCD dijet events with pile-up are shown in Figure 1.11. These can be parameterised as:

$$\langle E_T^{miss} \rangle \approx 1.25\sqrt{\Sigma E_T} \tag{1.5}$$

and

$$\sigma(E_T^{miss}) \approx 1.0\sqrt{\Sigma E_T} \tag{1.6}$$

## 1.2.5 The Muon Detectors

The Muon system provides measurements in the range $|\eta| < 2.4$ and is shown in Figure 1.12. Detectors are placed at four layers or stations in the barrel and endcap sections of the iron flux return. Three different types of gaseous detector are used due to the varying radiation and magnetic environments. The barrel section, covering $|\eta| < 1.2$, uses drift tubes (DT). The high muon and neutron background environment of the endcaps, $0.8 < |\eta| < 2.4$, means cathode strip chambers (CSC) are used instead of DTs. Resistive Plate Chambers (RPC) are used in both the barrel and part of the endcaps, $|\eta| < 2.1$.

The barrel region contains 250 chambers of up to 12 planes of drift tubes. The individual drift tubes have a cross section of $42 \times 13\,\mathrm{mm}^2$ and are filled with Ar and $CO_2$. Each drift tube consists of a central anode wire surrounded by aluminum cathodes. The induced charge has a maximum drift length of 2 cm or 400 ns. Each station provides a muon vector with a resolution of 100 $\mu$m in $\phi$ and 1 mrad in direction.

The two endcaps use 468 CSCs each of which is trapezoidal and contains 6 gas gaps. Each gap has a plane of radial cathode strips with perpendicular anode wires. The muon position is measured from the charge sharing of the radial cathode strips. Each station provides a muon vector with a resolution of $\sim$200 $\mu$m in $\phi$ and 10 mrad in direction.

The RPCs consist of a gas gap enclosed by two graphite-coated bakelite plates. The graphite forms a cathode with an aluminum strip used to read out the generated signal. The RPCs have a time resolution of $\sim$1 ns which makes them useful for identifying the bunch crossing time.

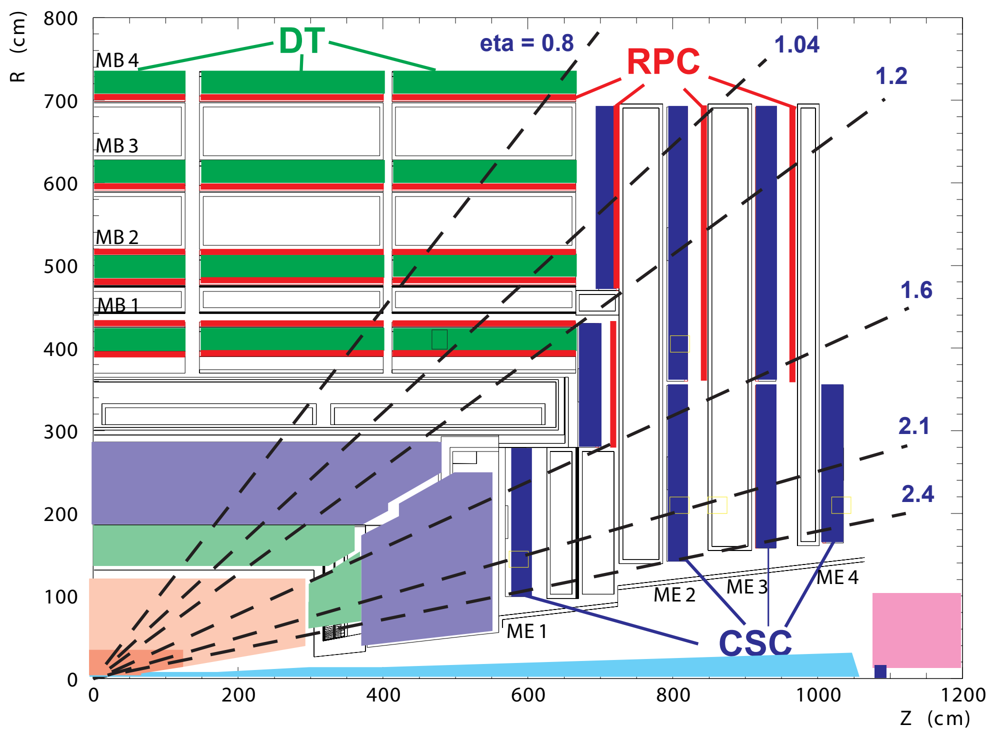


Figure 1.12: Layout of one quarter of the CMS muon system for initial low luminosity running. The RPC system is limited to $|\eta| < 1.6$ in the endcap, and for the CSC system only the inner ring of the ME4 chambers have been deployed. [9]

The muon system has a tracking efficiency of over 90% for 100 GeV muons. The resolution of the muon system, shown in Figure 1.13, is dominated by multiple scattering

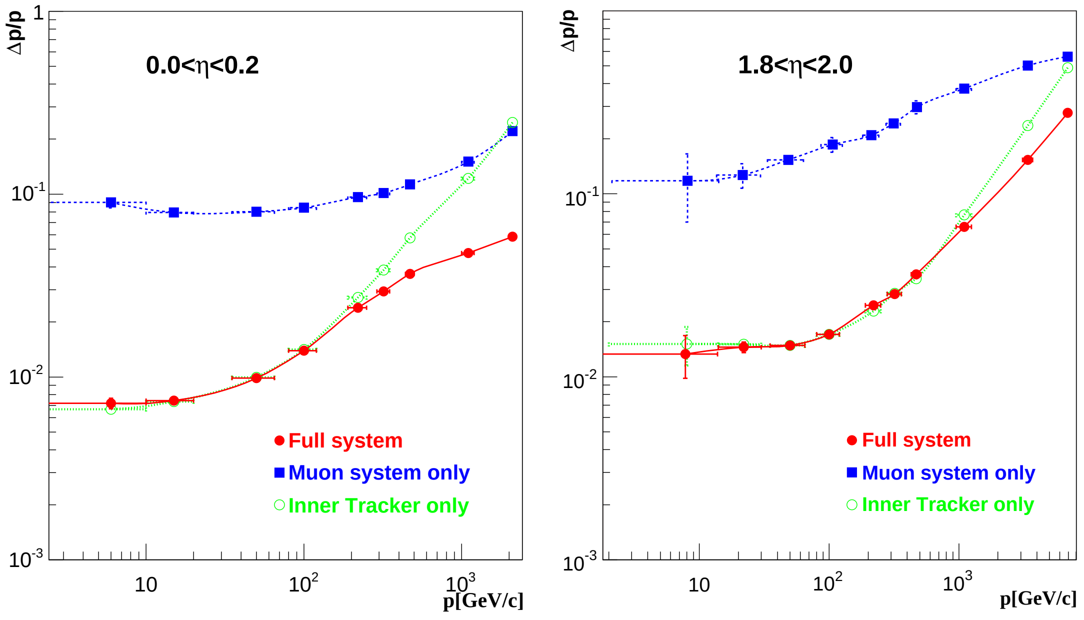


Figure 1.13: The muon momentum resolution versus $p$ using the muon system only, the inner tracker only, or both ("full system"). For the barrel, $|\eta| < 0.2$ (left) and endcap, $1.8 < |\eta| < 2.0$ (right). [9]

from the detector materiel in front of the muon detectors. At low momenta the resolution can be improved by using information from the tracker. At higher momenta multiple scattering can be neglected. The muons are then bent in equal and opposite directions before and after the coil, and this can be used at high energies to improve the resolution.

### 1.2.6 The Trigger System

At $\sqrt{s} = 14\,\text{TeV}$ the proton-proton inelastic cross section is roughly 100 mb. At design luminosity ($\mathcal{L} = 10^{34}\,\text{cm}^{-2}\,\text{s}^{-1}$) this results in $10^9$ inelastic interactions/s. Given the LHC bunch crossing interval of 25ns (40 MHz), this yields an average of 20 inelastic collisions per bunch crossing (event). Given that CMS has $\sim 10^8$ detector channels and one event comprises $\sim$ 1 MB of data this would give rise to a data rate of 100 TB/s. Storage systems place a limit of 100–200 MB/s, hence an online selection ("trigger") is used to reduce the event rate to $\sim$100 Hz.

The CMS trigger is split into two parts; a fast hardware (Level-1) trigger and a software High Level Trigger (HLT). The Level-1 trigger uses custom-built electronics hardware and reduces the event rate from 40 MHz to 100 kHz. The HLT runs on a commodity compute farm and reduces the rate further to $\sim$ 100 Hz for offline storage.

A period of 3.2 $\mu s$ is available for the Level-1 decision, but the signal transit time between the detector and the trigger hardware reduces this to less than 1 $\mu s$. The Level-1 trigger only uses calorimeter and muon information with the rest of the event stored in pipeline memory until the decision is reached.

The Level-1 trigger has separate calorimeter (regional and global) and muon triggers,

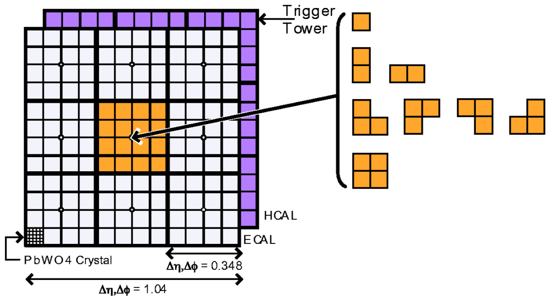


Figure 1.14: The Level-1 $\tau$ trigger algorithm showing the acceptable $\tau$-like shapes in the central region. [16]

that are all fed into a global trigger. The regional calorimeter trigger readouts both the ECAL and HCAL calorimetry in coarse grained samples of one HCAL tower or $5 \times 5$ ECAL barrel crystals, which corresponds to an area of $\Delta\eta \times \Delta\phi = 0.087 \times 0.087$. The regional calorimeter trigger identifies jet, photon and electron candidates (primitives).

The Level-1 jet trigger sums energy in a $4 \times 4$ array of towers which are then combined into a $3 \times 3$ array. Thus a jet is formed from a $12 \times 12$ array of towers corresponding to approximately a square unit in the $\eta - \phi$ plane. Separate lists are made of central and forward jets. The Level-1 $\tau$ algorithm (Figure 1.14), is similar to the jet algorithm but requires an isolated narrow energy deposit in the central $4 \times 4$ tower array.

The Global Calorimeter Trigger (GCT) combines the information from the regional calorimeter triggers and creates an $E_T$ ordered list of each primitive type. The global Level-1 trigger combines the information from the GCT and muon trigger and uses threshold cuts on the primitives to make the accept/reject decision.

Once accepted by the Level-1 trigger the event is readout and combined by an event builder via a switched network capable of a data transmission rate of 1 TB/s. Once combined the event is sent to a processor in the compute farm, which runs the HLT algorithms and makes the accept/reject decision. Scalability of the system is ensured with the addition of additional event builders and processor units which use the high speed network to communicate. As such during the low luminosity phase only a fraction of the total system will be implemented. As the data rate increases more event builders and processing nodes will be added.

By using a pure software high level trigger maximum flexibility of the reconstruction and triggering criteria can be achieved. The HLT rejects events as soon as possible by using the minimum of event information and by performing partial event reconstruction. Full granularity calorimeter information is used, followed by the tracker pixel detector and finally the full event is reconstructed using all detectors. If an event passes the HLT it is passed to the offline computing system together with a list of all primitives that passed the thresholds, the trigger bits.

# Chapter 2

# The Grid and the LCG

## 2.1 Introduction

The computing requirements for the LHC are unprecedented. By 2010 CMS alone will require over 60 PB of storage and 100 MSI2000 [17] of processing power [9]. At LHC inception no single computing centre was capable of providing these levels of resources. A mechanism for grouping the resources of multiple centres was required. The grid paradigm popularised by Ian Foster and Carl Kesselman [18] described such an architecture.

The grid was designed to provide an advanced computing infrastructure suitable for collaborative problem solving within science and engineering. Resources, both computational and storage, were shared amongst collaborating institutions within a dynamic Virtual Organisation (VO). The VO comprises individuals based at different institutions around the world all working towards a common goal. The ultimate aim of grid computing was to provide ubiquitous access to resources such that the user did not need to know where their work was carried out. They simply interacted with the grid and resources were provided. The grid was named by analogy with the power grid: users should consume computing power much as they consume electricity, without knowing the details of how or where it was generated. [1]

## 2.2 The LCG

By the year 2000 no appropriate large scale computational grid existed and so work on a custom-designed solution was started. This resulted in the European DataGrid (EDG) [20] project which was later succeeded by the Enabling Grids for E-sciencE (EGEE) project [21]. The LHC Computing Grid (LCG) [22] was originally based upon the EDG and later the EGEE implementations plus components from VDT [23], iVDGL [24], Griphyn [25] and Datatag [26]. [2] This chapter describes the EDG and LCG projects as they

[1]The concept of computing evolving into such a utility was first popularised by Martin Greenberger in 1964 [19].

[2]Other grid implementations existed i.e. Open Science Grid (OSG) in the USA and NorduGrid in Scandinavia and were used by the LHC experiments. Now they are part of the WLCG (Worldwide LCG).

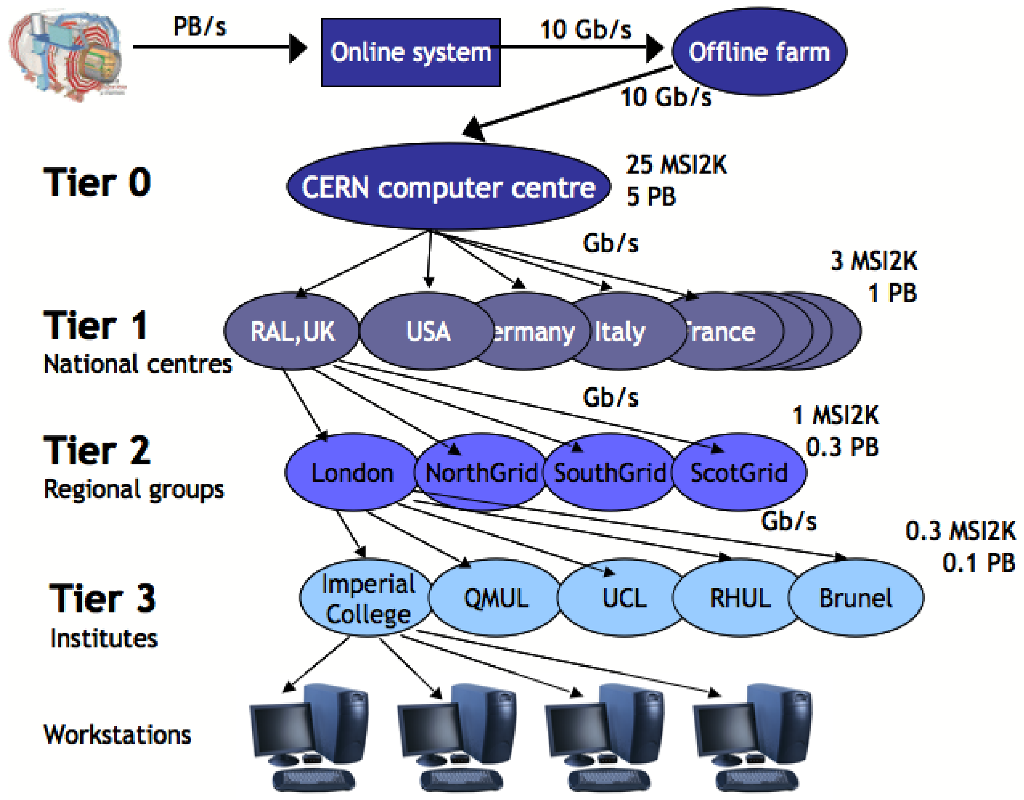


Figure 2.1: The MONARC tiered hierarchy. Network bandwidths are shown for inter-tier links as are computational and storage (disk only) resources for each site in a given Tier. From [27].

were in 2004–2005 when the work described in chapters 5 and 6 was carried out.

Within the LCG institutions were grouped into a hierarchy derived from the scheme devised by the Models of Networked Analysis at Regional Centres for LHC Experiments (MONARC) project [28]. This hierarchy took the form of a pyramid (illustrated in Figure 2.1) with CERN at the top level (Tier-0). The second level comprised large national computing centres (Tier-1's) followed by regional centres (Tier-2's) and smaller institutes (Tier-3). At each level the resources at a given site decreased, but the number of such sites increased to compensate, from 1 Tier-0 to ∼10 Tier-1's, ∼50 Tier-2's and an even larger number of Tier-3's.

To provide interaction between users and components a complex software (middleware) architecture was required, as can be seen in Figure 2.2. The User Interface (UI) was the gateway to the grid for users. The installed software allowed the user to manage data and submit computational jobs.

Data was stored on a site's Storage Element (SE) and recorded in a global file catalogue, known as the Replica Location Service (RLS) or Replica Catalogue. By using the client tools a user could copy data to/from and between SEs. Queries on the RLS were used for data discovery and location.

The Workload Management System (WMS) was provided by the Resource Broker

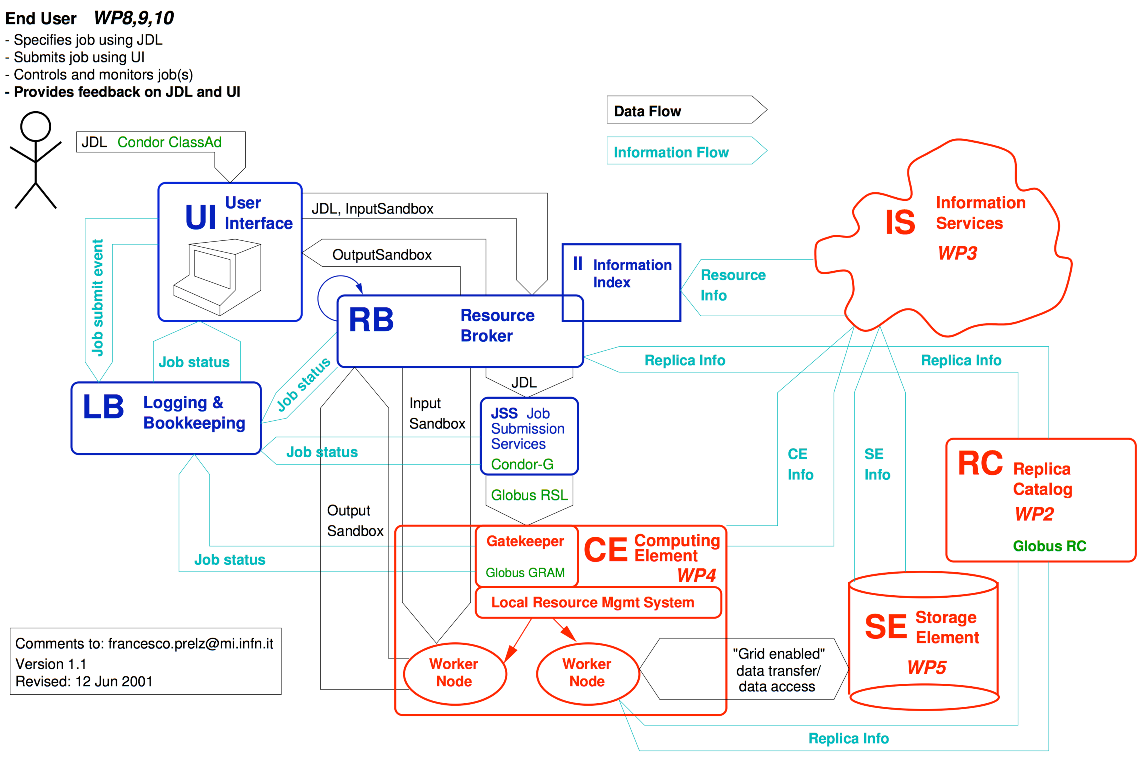


Figure 2.2: An overview of the major LCG components. From [29]

(RB). The RB's role was to accept users' jobs and take responsibility for assigning them to a site for processing. The Computing Element (CE) controlled the processing at a site, receiving jobs from the RB and scheduling them for execution on the site's Worker Nodes (WNs). The Logging and Bookkeeping (LB) system recorded changes in job state and was queried by users to determine the state of their jobs.

The Information Services (IS) listed all the RBs, CEs and SEs and was used by each system, and end users, to discover and obtain information about the other components.

### 2.2.1 Data Management

The LCG data management system was specifically designed to cope with the types of data produced by particle physics experiments: i.e. large amounts of read-only data. It had also been designed to provide high levels of data availability and integrity with multiple distributed file replicas. To facilitate this the RLS provided the concepts of Logical File Name (LFN), the canonical name by which a file was known, and Physical File Name (PFN), a name referring to a specific file instance. A user would refer to a file by its LFN, the RLS would resolve the best replica and this instance was used in the data operation. As well as holding the LFN/PFN mapping the RLS could also contain a limited number, O(10), file metadata fields. It was generally assumed that files within the RLS were read-only so synchronisation between replicas was not supported.

### 2.2.2 Workload Management

To submit a job the user first had to describe their job in a Job Description Language (JDL) file [30]. This file was written in the Classified Advertisements (ClassAds) [31] syntax and listed the job structure and attributes. Fields included information such as executable name and arguments. It was also possible to specify requirements for the job which were used by the RB to determine an appropriate site to run the job. Typical requirements included criteria such as operating system, site capacity and processing time limits.

In the JDL the user could also specify input and output files. There were two mechanisms for file handling: small files (e.g. log files and text files) were sent with the job whereas large data files were saved to SEs and registered with the RLS. Input (output) files sent with the job were said to be part of the input (output) sandbox. As well as containing the user-defined input files, the input sandbox contained the users executable and an optional file used as standard input for the job. The output sandbox contained the user defined output files and the standard output and error streams from the job.

Once the user had a JDL file they could submit the job via the software on the UI to an RB. The RB received both the job and the input sandbox then selected the most appropriate CE taking into account the specified requirements, a process known as match-making. Once a CE received the job it was added to the local batch queue and scheduled for execution. Before the job started, the WN downloaded the input sandbox from the RB. The user's job was then executed. Once the job had finished the output sandbox was transferred to the RB. Changes in job state were recorded to the LB. This information was then available to the user. The output sandbox was retrievable from the UI.

The WMS interacted with the RLS when the user specified one or more LFNs in the JDL. It was LCG policy to send jobs only to sites which hosted all of the required data. Thus, if a user specified any input LFNs, the RB contacted the RLS, retrieved the list of hosting sites and only considered those in the match-making decision. At the end of the job it could copy any output files to an SE and register them with a user-provided LFN.

### 2.2.3 LCG object persistency

As well as core grid functionality the LCG project developed related software services for the LHC experiments. One of these was POOL (Pool Of persistent Objects for LHC) [32] which provided object persistency and file input/output (IO). This system was used by 3 of the LHC experiments, including CMS, in their event data model.

POOL used the LFN/PFN convention and required a file catalogue to store these. This catalogue was used to locate files when users referred to them with an LFN. Multiple file catalogue implementations existed including the RLS, MySQL [33] databases and XML files. The user had to provide a contact string for a POOL file catalogue before requesting any file operations. POOL provided an Application Programming Interface (API) for queries of the file catalogue. This API allowed queries of LFNs, PFNs and metadata as compared to the native RLS API that was restricted to queries of LFN-PFN only.

When an application used POOL for writing to a file it was automatically added to the file catalogue. When a file was accessed the application/user used the LFN and POOL would consult the catalogue to locate a physical replica to use.

This architecture did not make applications grid-aware, however, because files could only be accessed within a site. POOL had no mechanism for resolving the "best" PFN or for trying multiple replicas. If a PFN referring to a file hosted elsewhere was returned it would cause the application to fail.

## 2.3 Summary

The LHC computing requirements were greater than any single site could provide. Therefore the LCG was developed to provide a distributed computing infrastructure for the LHC experiments. This infrastructure provided both data and workload management. However these were not seamless and did not provide all of the necessary functionality. If required, more advanced and custom features had to be implemented by each experiment on top of the LCG. The system developed for CMS is described in the next chapter.

# Chapter 3

# CMS Computing Model

## 3.1 Introduction

[1] The extensive scope of the computing requirements at each LHC experiment were outlined in the previous chapter as was the need for CMS to develop its own computing infrastructure on top of the LCG. This chapter details the computing model chosen by CMS.

The CMS computing model lists the 2010 requirements as 116.6 MSI2000 [17], 34 PB of disk and 59 PB of tape [34]. In order to manage these resources effectively a highly developed and scalable computing model was needed. This model had to be capable of managing sufficient levels of resources while facilitating complex particle physics workflows for the non-expert user.

Not all LCG institutions support CMS. Those that do include a large Tier-0 centre at CERN, 7 Tier-1's, ∼25 Tier-2's and many more Tier-3 centres. CMS also requires an analysis facility at CERN to be available for analysis activities close to the experiment, termed the CMS CERN Analysis Facility (CAF).

It is likely that the first years of CMS running will be characterised by a poorly-understood beam and detector. As understanding of the beam and detector evolves frequent data reprocessing will be required. During this time LHC operation will be inconsistent and the data flow from the detector will be erratic. Even so there will be great pressure to discover new physics as rapidly as possible, which will require full resource utilisation and the ability to prioritise critical activities.

Due to the expected constraints it has been decided that the computing model, during the startup phase, should be simple, reliable and give the best possible environment for early physics discoveries. The baseline computing model emphasises:

- Fast and frequent reconstruction;
- Streamed primary datasets, with distribution and processing driven by priorities;

[1] This review has largely been taken from [34] and [9].

- Joint distribution of raw and reconstructed data, allowing wide distribution and direct comparisons;
- Multiple compact data formats with multiple distributed copies; and
- An effective and efficient bookkeeping system.

There are also plans for beyond-the-baseline architectures and services, which will be implemented once the experiment has stabilised and greater functionality is required. These are not discussed in this thesis.

CMS will operate a structured analysis environment with analysis groups focusing on well-defined objectives whose priority has been determined by the collaboration. Computing resources will be prioritised for different activities; policies at Tier-0 and Tier-1 centres will be set to meet analysis group requirements, whereas controls at Tier-2 and Tier-3 centres will be looser. These controls will be especially important at start-up when only limited resources may be available.

## 3.2 Data Formats

A canonical Tier-1 lacks the storage capacity for a complete copy of the raw data, so to maximise data availability more compact data formats are to be utilised. This will allow data to be made available for analysis at multiple sites. The formats (data tiers) used are:

- RAW. This is the output format of the HLT farm. It contains data from all detector channels as well as the Level 1 and High Level Trigger bits. The target event size is 1.5 MB;
- RECO. This is the output from event reconstruction of the RAW data, where, objects (tracks, vertices etc.) have been reconstructed and calibrations applied. The target event size is 0.25 MB;
- AOD (Analysis Object Data). This is the output from further reconstruction (re-processing) of RECO data. Physical objects (electron, photons etc.) have been reconstructed and calibrations applied. This is expected to be the primary data used in analysis. The target event size is 0.05 MB; and
- TAG. This format indexes other event data and is the output from skimming of event data. This process is used to identify events with similar characteristics. This format has not been implemented to date.

When data is collected/produced it is grouped with similar data into datasets. This grouping is driven by the HLT bits. The same dataset may have RAW, RECO and AOD data tiers, therefore both must be specified to identify data uniquely.

| | | Running Year | | | | |
|---|---|---|---|---|---|---|
| | | 2007 | 2008 | 2009 | 2010 | |
| Conditions | | Pilot | 2E33+HI | 2E33+HI | E34+HI | |
| Tier-0 | CPU | 2.3 | 4.6 | 6.9 | 11.5 | MSi2k |
| | Disk | 0.1 | 0.4 | 0.4 | 0.6 | PB |
| | Tape | 1.1 | 4.9 | 9 | 12 | PB |
| | WAN | 3 | 5 | 8 | 12 | Gb/s |
| A Tier-1 | CPU | 1.3 | 2.5 | 3.5 | 6.8 | MSi2k |
| | Disk | 0.3 | 1.2 | 1.7 | 2.6 | PB |
| | Tape | 0.6 | 2.8 | 4.9 | 7.0 | PB |
| | WAN | 3.6 | 7.2 | 10.7 | 16.1 | Gb/s |
| Sum Tier-1 | CPU | 7.6 | 15.2 | 20.7 | 40.7 | MSi2k |
| | Disk | 2.1 | 7.0 | 10.5 | 15.7 | PB |
| | Tape | 3.8 | 16.7 | 29.5 | 42.3 | PB |
| A Tier-2 | CPU | 0.4 | 0.9 | 1.4 | 2.3 | MSi2k |
| | Disk | 0.1 | 0.2 | 0.4 | 0.7 | PB |
| | WAN | 0.3 | 0.6 | 0.8 | 1.3 | Gb/s |
| Sum Tier-2 | CPU | 9.6 | 19.3 | 32.3 | 51.6 | MSi2k |
| | Disk | 1.5 | 4.9 | 9.8 | 14.7 | PB |
| CMS CERN | CPU | 2.4 | 4.8 | 7.3 | 12.9 | MSi2k |
| Analysis Facility | Disk | 0.5 | 1.5 | 2.5 | 3.7 | PB |
| (CMS-CAF) | Tape | 0.4 | 1.9 | 3.3 | 4.8 | PB |
| | WAN | 0.3 | 5.7 | 8.5 | 12.7 | Gb/s |
| Total | CPU | 21.9 | 43.8 | 67.2 | 116.6 | MSi2k |
| | Disk | 4.1 | 13.8 | 23.2 | 34.7 | PB |
| | Tape | 5.4 | 23.4 | 41.5 | 59.5 | PB |

Table 3.1: Time Profile of CMS Computing Requirements. [9] These requirements change frequently, see [35] for up to date figures.

## 3.3 Tier roles & responsibilities

CMS has assigned each tier within the LCG MONARC hierarchy roles according to their resource levels. The required computing resources at each Tier are listed in Table 3.1.

### 3.3.1 Tier-0

The responsibility of the Tier-0 is to accept data from the CMS online system (HLT). This is then archived, reconstructed and a copy distributed to the Tier-1's. It is essential that this proceeds with minimum delay to avoid a build up of data on the input buffers. Thus the resources of the Tier-0 must be dedicated and continually available. During periods that CMS is not taking data the Tier-0 will be used for heavy-ion reconstruction, which takes ~10-50 times more processing than that of a proton-proton event.

### 3.3.2 Tier-1

The Tier-1's have a number of responsibilities including secure data storage and centrally co-ordinated processing. The Tier-1's as a whole are responsible for storing the custodial copy of the RAW data with the Tier-0 copy only accessed in the event of data loss. Due to its small size each Tier-1 stores the entire AOD. Both real and Monte Carlo data is included in the Tier-1 storage.

Tier-1 data processing includes reprocessing, skimming and a small amount of analysis. A Tier-1 will reprocess all of its data at least twice a year. A limited amount of well managed analysis will be allowed, probably making use of the RAW data. All activity at the Tier-1 will be managed and subject to CMS control and policies.

### 3.3.3 Tier-2

The main role of the Tier-2 is to support a portion of the physics community and run analysis jobs. Many detector-specific calibration and analysis groups are expected to operate at a Tier-2 “close” to the relevant experts. It is expected that such a Tier-2 would provide extra resources for that group, i.e. extra storage space and relevant software installations. The analysis will originate from both local and remote users.

A nominal Tier-2 has enough disk space (no tape is required at Tier-2’s) to hold approximately one-tenth of the current RECO data and half of the current AOD. Tier-2’s have no custodial responsibility for data. Data will be downloaded, analysed and periodically replaced. As well as providing for analysis the Tier-2’s must provide processing capacity for Monte Carlo production ($\sim 10^8$ events per year). These data will be stored at the Tier-1.

### 3.3.4 Tier-3

In CMS, Tier-3 sites are users’ home institutes, where they are provided with the facility to develop and submit their analysis. The major usage pattern consists of users submitting analysis jobs either to run locally or at a remote site and for the results to be returned to them at the Tier-3. Some Tier-2’s will also provide Tier-3-type facilities as well as fulfilling their nominal role as defined by the computing model.

### 3.3.5 CMS-CAF

The CMS-CAF provides general facilities for the whole collaboration, serves users without a local Tier-1/2 and provides facilities for low latency, experiment-critical activities. General services include user logins, database support, production bookkeeping and software repositories. High priority work, generally taking advantage of the only copy of all RAW data, will include:

- Detector diagnostics - particularly useful during early running stages and after periodic shutdowns;
- Trigger performance studies - i.e. optimisation and algorithm development. These studies will be carried out in response to changing understanding of the detector or the need to focus on specific channels; and
- Calibration and alignment - to support the HLT and to be used in the (re-)reconstruction at the (Tier-1) Tier-0.

The analysis facilities offered to users will be approximately equivalent to 2 nominal Tier-2 centres. However, it is essential that individual users cannot interfere with the critical work undertaken here, hence strict policies and quotas will be put in place.

## 3.4 Data Distribution

The data distribution system must be capable of multiple simultaneous transfers between sites while maximising network usage and ensuring data integrity. Transfers must be automatic, provide for error recovery and conform to priorities determined by the experiment. It is vital that data from the detector be archived and streamed to the Tier-1's with minimum delay and without data loss. Lower priority transfers include data for user analysis being streamed to the Tier-2's. Monte Carlo data generated at the Tier-2/3's must also be moved to the Tier-1 for long-term storage.

The planned data flow for CMS is illustrated in Figure 3.1. The online system will be set to write out events at the maximum rate that the offline computing model can support, which ties the physics reach of the experiment to the performance of the offline computing system. In the baseline model this rate will be set to a minimum of 225 MB/s giving an event rate of 150 Hz.

If an event passes the HLT it is passed to the offline system. The Tier-0 archives a copy of this data then passes it through a first pass reconstruction that results in RAW, RECO and AOD data products.

Each Tier-1 receives a share of the RAW and RECO data consistent with their resources and a complete copy of the AOD. Periodically the Tier-1 will reconstruct its RAW data to produce new RECO data and then reprocess that to produce new AOD.

The Tier-2's obtain a share of the RECO data and half of the AOD from the Tier-1. The specific content will vary as physics priorities change and Tier-1 processing progresses. As such, a Tier-2 is expected to be capable of refreshing all of its data within 30 days, corresponding to a data import rate of 5 TB/day.

## 3.5 Computing Services

CMS has developed the following guiding principles for its computing services:

- Optimise for read access. In general CMS data will follow the general HEP pattern of being created, never modified and subsequently read many times;
- Optimise for the bulk case, while still allowing basic tasks. In CMS the large amounts of data and jobs will require operations on a large scale i.e. the data transfer will be managed at the dataset level. However the functionality still exists for users to copy individual files;
- Minimise dependencies of processes on the worker nodes (WNs). CMS expects $10^3 - 10^4$ worker nodes to be in constant use. This presents a large reliability problem as

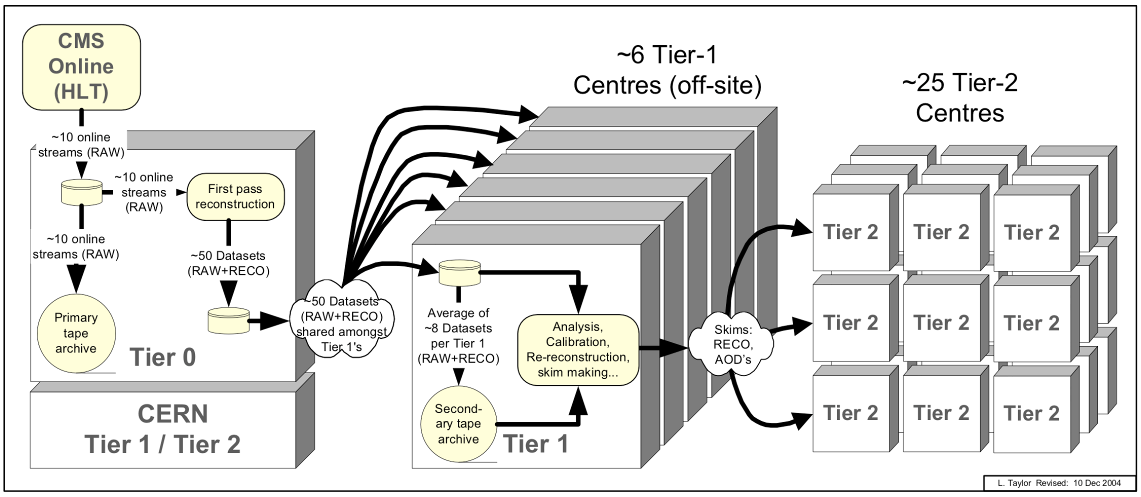

CMS Online (HLT)
~10 online streams (RAW)
~10 online streams (RAW)
First pass reconstruction
~10 online streams (RAW)
~50 Datasets (RAW+RECO)
Primary tape archive
Tier 0
CERN Tier 1 / Tier 2
~50 Datasets (RAW+RECO) shared amongst Tier 1's
~6 Tier-1 Centres (off-site)
Average of ~8 Datasets per Tier 1 (RAW+RECO)
Analysis, Calibration, Re-reconstruction, skim making...
Second-ary tape archive
Tier 1
Skims: RECO, AOD's
~25 Tier-2 Centres
Tier 2
L. Taylor Revised: 10 Dec 2004

Figure 3.1: Schematic flow of bulk (real) event data in the CMS Computing Model. Not all connections are shown, e.g. peer-to-peer connections between Tier-1's and Monte Carlo data transfer from the Tier-2 to Tier-1. [9]

nodes fail or suffer outages. The WN is made more reliable by removing as many external dependencies as possible. Any dependencies that do remain should be local to a site to avoid any single point of failure for the entire system;

- Allow provenance tracking. It is a requirement of the software and computing frameworks to track the provenance of datasets. This includes run-time parameter sets and software versions;

- Site configuration information should remain local to the site. This allows system administrators to setup their site however they wish while allowing CMS applications to discover any information required; and

- Keep the solution simple. The start-up of CMS is likely to be a hectic and confusing time, thus it has been decided that the computing system should be as simple as possible while providing all necessary functionality.

The CMS computing system consists of a distributed set of systems and services. These services consist of a mix of generic grid, site-specific and CMS-specific components. Figure 3.2 shows the major components of the computing system. The main systems are:

- Data Management System - the CMS data management and movement tools.

- Grid Workload Management System - the core grid systems and services of which CMS makes use.

These pieces are tied together by the CMS Workflow Management (WM) system. This system supports all necessary workflows in CMS, including (re-)reconstruction, reprocessing, calibration, MC production, skimming and user analysis, while shielding users from the complexity and implementation details of the underlying components.

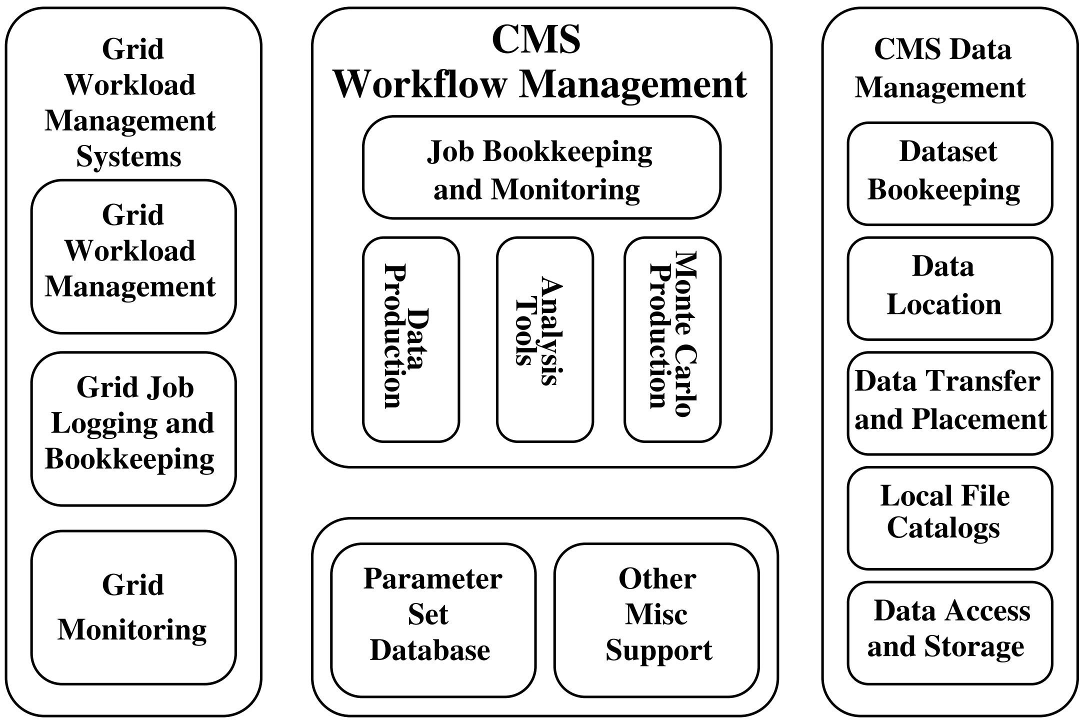


Figure 3.2: Overview of systems and services supporting the CMS workflow management system. [9]

## 3.6 Data Management System

The CMS Data Management (DM) system is designed to provide necessary data management functionality for CMS, i.e. allowing physicists to discover, access and transfer various forms of data. These facilities will be used in situations ranging from organised large-scale data placement operations to individual user file transfers. The system must be suitable in all situations.

In the baseline model the experiment data placement will be pre-determined leaving the CMS WM system to steer jobs to the correct location. The DM system will provide functionality for the WM system to accomplish this.

The Data Management system understands both user-oriented terms such as dataset as well as technical details such as file names that need not necessarily be exposed to the user. An "event collection" is defined as the smallest data unit that may be selected. A dataset is defined as a set of "event collections" that share a set of trigger bits. The dataset concept is used by physicists to specify data for their analysis jobs, but the jobs will be configured with, and at run time refer to, event collections. Event collections are stored in files that are themselves grouped into "file blocks". These file blocks are groups of files that are generally distributed and accessed together.

The DM architecture is based on a loosely-coupled set of components, which together provide the functionality required of the system. The basic components of the system are:

- Dataset Bookkeeping System (DBS) - lists the available data. For the physicist this will be the primary means of data discovery. The DBS can be queried with criteria such as run range, data tier, software version and data quality flags. Results will be returned as either datasets, event collections or file blocks. These can then be used to configure the CMS analysis framework;

- Data Location Service (DLS) - maps file blocks to sites. Queries will result in a list of sites that contain the given data;
- Data Placement and Transfer System - handles file transfers. The data placement system will be used to manage data movement and placement. File blocks will be subscribed to sites and the request passed on to the data transfer system. The data transfer system will handle reliable end-to-end transfers of individual files;
- Local File Catalogue - provides file lookups at a site. This catalogue presents a POOL interface that is able to return file locations at a site and can be used by CMS applications to locate event data. In the current implementation this is a text file that contains lookup rules to determine the correct file location; and
- Data Access and Storage System - provides access to files and manages the local mass storage system, if any.

## 3.7 CMS Workflow Management System

### 3.7.1 Introduction

The CMS Workflow Management (CMS WM) System combines the CMS DM and grid workload management systems and presents them in a uniform manner to the user. The WM system includes applications designed to support all major CMS workflows including prompt reconstruction, prompt calibration, re-reconstruction, reprocessing, calibration, Monte Carlo production and analysis.

The primary unit of work in the WM system is a task. The task contains an application, configuration parameters and optionally a data selection. The DBS and DLS search for the data selection and once the WM system discovers the nature of the data to run over, i.e. number of runs/events, it can decide how best to run the workflow. In general, the task will be split into many jobs each running over a subset of the total requested data. Once the individual jobs have been created they are submitted to a supported grid WMS. This then takes responsibility for scheduling the job at an appropriate site, i.e. one that contains the required data. Once the job is finished the CMS WM system will retrieve the results and handle any output files.

Figure 3.3 shows how the CMS DM system and the grid WMS are combined. The UI is the gateway to both the grid and CMS WM system with both CMS WM applications and grid tools installed. Individual CEs and SEs in supported grid middlewares advertise themselves via their native information services. The CMS WM system can then run a given workflow on the most appropriate resources on any supported grid middleware.

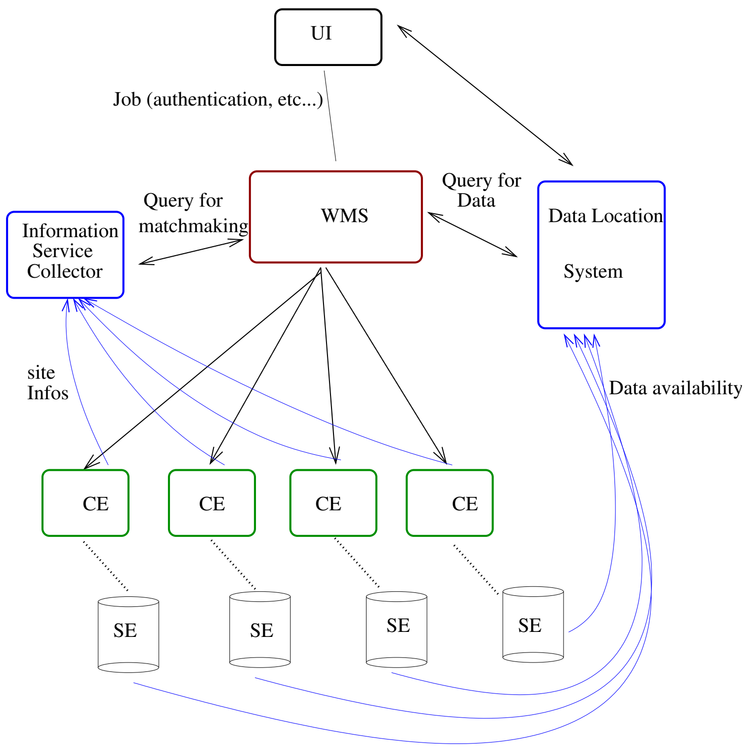


Figure 3.3: The baseline CMS WM system architecture. [9]

The work of the CMS WM system is simplified for workflows that run exclusively on the Tier-0 (prompt calibration and reconstruction) as no grid middleware is needed.

### 3.7.2 User analysis

One of the prime supported CMS workflows is that of user analysis. These analyses are carried out by individuals or groups of physicists and will be the source of physics discovery at CMS. All data taken by CMS will be analysed many times by different groups, each looking for different physics. Classic CMS analysis, described below, limits the user to running over local data and requires the user to handle job creation, submission and output file handling. The CMS WM expands on this by providing full automation from analysis creation through to job splitting and output file handling.

#### Classic CMS Data Analysis

Analysis of event data is performed with the CMS analysis framework. This provides a layer of common functionality (event access, reconstruction algorithms, calibrations etc.) that users may call on in their analysis programs. Users build their code against this framework and an executable is produced. This executable dynamically links with the framework libraries as extra functionality is required. User options, datasets and other parameters are passed via a configuration file. In CMS the standard practice is for the user's analysis code to produce an ntuple file that is then analysed with ROOT [36], or similar a tool, to produce final plots.

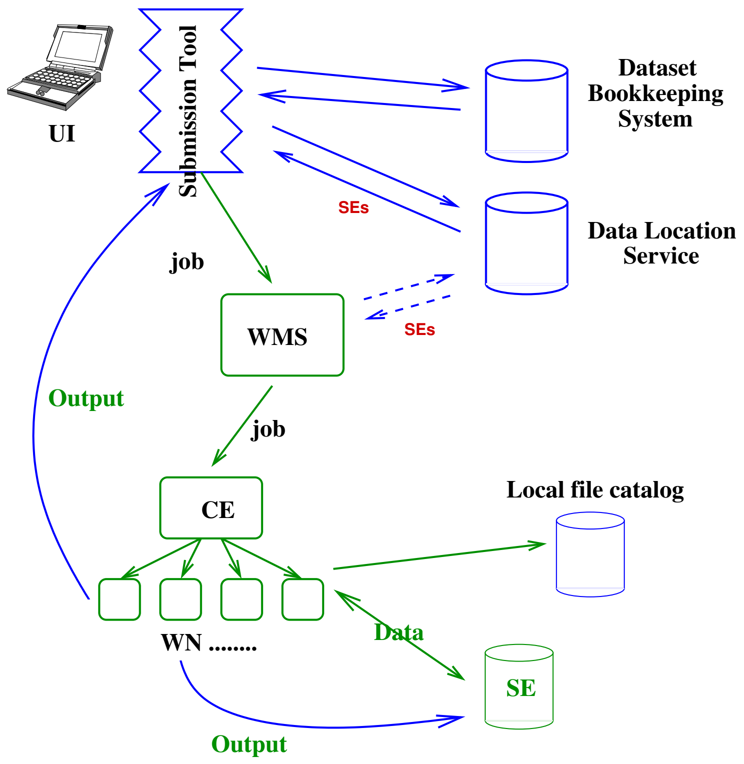


Figure 3.4: Distributed data analysis with the CMS WM system. [9]

The framework utilises POOL for file IO and catalogue functionality. The CMS analysis framework refers to data files via their LFNs and relies on POOL for file location and IO. This means that to run in the framework, the user has to specify, in their configuration file, at least one POOL catalogue that contains the required event files. The framework has no knowledge of the grid as it runs locally and assumes that files in the POOL catalogue are accessible. Users can only therefore analyse data at their local site, which is clearly incompatible with CMS's distributed data model. The CMS Workflow Management System solves this problem and provides the user with access to all CMS data and computing resources.

### Distributed User Analysis Workflow

The classic analysis scenario is useful for developing analysis code and analysing small numbers of events. However most CMS analyses require running over thousands or millions of events. To do this a user has to write their own machinery to create possibly hundreds of analysis jobs then submit them and finally handle the hundreds of resulting output files. This process is tedious and prone to error.

A main function of the CMS WM system is to simplify and automate these functions for the user. Functionality includes data verification, data location, job splitting, packaging, submission, output retrieval and merging. This workflow is illustrated in Figure 3.4 and presented below.

The user begins (as in the "classic case") with an analysis application they wish to run, various configuration options and the data they wish to run over. These data may either be a named dataset or determined from a list of requirements i.e. software versions and HLT flags. The DBS is then contacted and the required data determined. This query results in a list of available event collections (and file blocks). The WM tool is then able to decide how best to divide up the user's analysis task into multiple jobs, each of which accesses a subset of the total data. The job splitting is driven by the user's configuration (maximum events per job, desired job run time etc.).

The CMS WM system can then create the configuration for the jobs. This is a two step process as each job is configured at the CMS application level and at the grid WMS level. The application configuration requires the input event collections. The grid WMS configuration needs information in order to steer the job to a site that holds the appropriate event collections. This is accomplished by contacting the DLS to discover which storage elements (SEs) contain the required data. The user may specify that the output be returned directly to them or, if the output is likely to be large or accessed by others, copied to a remote storage element.

The CMS WM system submits, and tracks jobs to the WMS, or local site batch system. Jobs submitted to distributed resources are sent with a list of SEs that contain the requested data blocks and it is the task of the grid WMS to steer the job to the "best" site in terms of load, data access cost etc.

Once a job arrives at a site and locates the site-specific information, the user's application is presented with an environment that is identical to the development setup. This includes such things as CMS software installations and access to the CMS conditions DB. Once all of these have been found, the user's application runs exactly as during development with files streamed from the local storage system by use of the local file catalogue.

The user tracks their jobs using the CMS WM job logging and bookkeeping system, which uses the WMS information services as an information source. Optional real-time monitoring allows the user to monitor the progress of the job including such information as event number and CPU usage. This is optional as it requires the site to allow information to be sent from the job to an external service. The user may also find this unnecessary, especially if they have a large number of active jobs. Once the job finishes the output is either saved to an SE or returned via the WMS to the user's filesystem. The output from many jobs may be merged to aid the users further analysis of it.

## 3.8 Alternative computing models within HEP

Within HEP the computing requirements of experiments vary considerably. Until recently all computing models relied on largely centralised resources. However the larger experiments are now restructuring their models to take advantage of grid computing.

### 3.8.1 Tevatron Experiments

The highest energy collider currently in operation is the Tevatron, based at Fermilab. The experiments based here (CDF [37] and D0 [38]) have some of the most demanding computing requirements of any HEP experiment to date. Together they require $\sim 7\,\text{PB}$ of tape storage by 2007 [39]. Initially the computing models for both experiments relied on centralised resources with large computing farms at Fermilab. However, both are now taking advantage of significant computing resources outside of Fermilab.

To facilitate this move to distributed computing Fermilab has developed some common grid services. Sequential Access via Metadata (SAM) is a data handling system that provides a mechanism for accessing distributed data. Data is transparently copied from remote sites on demand. The JIM (Job and Information Monitoring) services provide a mechanism for submitting SAM jobs to remote resources and then monitoring these jobs. The combination of SAM and JIM (termed SAMgrid) provides users with access to all available data and computing resources. Currently D0 uses SAMgrid while CDF uses SAM but not JIM, its submission software handles job submission via Condor and LCG tools.

Both experiments have significantly expanded their available resources by using distributed resources. Around 45% of CDF's analysis capacity is located at remote sites and all D0's reprocessing and Monte Carlo generation has been completed offsite, as in the CMS computing model [39]. Both experiments are actively looking for ways to meet as much of their processing needs as possible remotely. Fermilab has developed their own grid technologies rather than join the LCG project, although a new project has been created aimed at allowing LCG and SAMgrid to interoperate.

### 3.8.2 LHC experiments

The computing models of the LHC experiments (ALICE, ATLAS, CMS and LHCb) were all designed after the development of the grid paradigm and therefore all utilise distributed computing. Each experiment's computing requirements were similar resulting in comparable computing models. The LCG project was specifically designed to provide a common grid layer for the LHC experiments, although each experiment makes use of it in a slightly different way.

ATLAS and CMS, the two general-purpose detectors, have similar computational requirements [22]. For planning purposes it is often assumed that these are equal but there are significant differences. An example of this is the amount of simulated MC data required; ATLAS will produce MC data equivalent to 20% of their accumulated real data whereas CMS requires a ratio of 100%. Another difference between the two is created by the composition of the two collaborations, which has resulted in more Tier-1 sites available to ATLAS than to CMS. To compensate for this CMS will rely heavily on resources at CERN (Tier-0 and CMS CAF) and the Tier-2's. The activities planned by each experiment for each Tier are similar with both models using the Tier-1's for re-reconstruction, reprocessing and skimming while the Tier-2's are responsible for user analysis and MC production.

ATLAS and CMS are also similar in their use of LCG software. Both have taken the WMS as given and written client-facing software that wraps and aggregates the LCG commands to provide higher level functionality and greater ease of use [40,41]. For data management both experiments found LCG tools lacking in high level management functionality and therefore created their own systems that managed data flows between sites using the LCG tools [42,43].

ALICE specialises in heavy ion physics, where events are more complex than proton - proton interactions but run at a much lower luminosity. By 2010 ALICE requires approximately half of the processing power and a third of the mass storage of CMS [22]. The computing model, built to satisfy these needs, is very similar to the model used by ATLAS and CMS. Both have similar usage patterns at each of the tiers, the main difference being the scale of the activity.

In contrast to ATLAS and CMS the ALICE collaboration have created their own distributed computing framework, known as AliEn (Alice Environment) [44]. AliEn was developed at the same time as the LCG and was designed to provide transparent access to ALICE's homogeneous distributed resources. AliEn runs on ALICE resources and consists of central and site agents interfacing via SOAP web services.

For workload management the main difference between AliEn and the LCG is that AliEn has a single task queue for all jobs whereas LCG has multiple RBs each managing jobs independently. Another difference between the two is the method used to distribute jobs; LCG uses the "push" method where ALiEn uses the "pull" method. In AliEn, when a job slot becomes free at a site, a local agent contacts the central AliEn task queue and obtains a suitable job. In LCG the RB assigns jobs to sites according to the load in the system. There are advantages and disadvantages to each approach, e.g. multiple RBs have no single point of failure and can perform load balancing over the entire grid whereas a global job queue enables experiment-wide scheduling policies to be enforced.

Sites are reluctant to run VO-specific services due to the arbitrary hardware required, high levels of support and security concerns. Thus LCG looked for a way to normalise these services and developed the VO box concept. This was a grid node that was designed to run VO specific services for a site with remote management. Thus the site only needed to provide a standard grid node and each VO could organise their own services. These were used by ALICE to run the AliEn site services and were required at both Tier-1 and 2. However a number of Tier-2's were reluctant to run VO boxes and thus did not fully support ALICE.

LHCb specialises in B physics. LHCb's computing requirements are relatively modest: by 2010 it requires approximately a sixth of the processing power and a fifth of the mass storage of CMS [22]. LHCb has adopted a computing model substantially different from that of the other experiments. Due to the relatively modest processing requirements needed for reconstruction, reprocessing and skimming, enough capacity will remain at the

Tier-1's for all analysis activity. This is in contrast to the other experiments where only limited, managed, analysis will be permitted on the Tier-1's. This has the advantages that only ∼6 sites are required to provide facilities for analysis, massively reducing the management overhead and Tier-1 to Tier-2 network traffic. However, user analysis is seen as a source of chaotic behaviour that must be effectively controlled otherwise it may interfere with the high priority managed workflows. The only role for Tier-2 and Tier-3 LHCb sites is MC production thus the level of storage required at these sites is low.

To implement their computing model LHCb have created their own framework, known as DIRAC (Distributed Infrastructure with Remote Agent Control) [45]. DIRAC consists of remote agents communicating via the XML-RPC protocol. A typical DIRAC workflow is similar to the approach of AliEn, with agents at each site pulling in jobs from a central task queue. To integrate with other grid middleware implementations, e.g. LCG, DIRAC has developed the "pilot" job. A central agent exists that submits jobs to each grid's native WMS that, when executed, contacts the DIRAC task queue and obtains the real user's job.

All of the LHC experiments rely on distributed computing. The more computationally intensive experiments (ALICE, ATLAS and CMS) have all adopted similar computing models. Reprocessing, skimming and limited analysis will be performed at the Tier-1's with the majority of user analysis conducted at the Tier-2's. All Monte Carlo (MC) will be generated at the Tier-2/3's. LHCb, due to its relatively modest requirements can handle all its non Monte Carlo processing at the Tier-1's. This greatly simplifies the system but the resulting effect on the Tier-1's is not fully understood.

Both ATLAS and CMS rely on LCG to provide basic grid functionality with their custom applications handling the complex workflows. Both ALICE and LHCb have developed their own grid middleware layers. LHCb is integrating their software into LCG while ALICE is keeping a large fraction of their custom grid software.

## 3.9 Summary

The scale and complexity of CMS computing required a well designed computing model. This model is based on the LCG and provides increased functionality and ease of use. This functionality is divided into two main components: the Data and Workflow Management systems. One of the prime aims of the Workflow Management system is to support distributed user analysis. This is a complex application that had to bridge many systems and as such a number of prototypes have been developed. One of these is described in the next chapter.

# Chapter 4

# Distributed analysis with GROSS

## 4.1 Introduction

GROSS (GRidified Orca Submission System) [1] was created to provide a reliable method for submitting CMS analysis jobs to experiment resources worldwide. It was designed to perform all of the major operations required by physicists when submitting large analysis tasks, i.e. once the analysis code had been developed and an analysis of large amounts of data was required. The user's analysis task was split into smaller, independent jobs which were submitted to a batch system. These were monitored and any output retrieved.

To guide the design of the CMS Computing Model a series of increasingly complex data challenges were run. One of the major challenges was CMS Data Challenge 2004 (DC04) [46]. A principal aim of this challenge was to perform distributed analysis, and it was this role that GROSS was designed to fill. At the time users generally used local batch resources therefore it was decided that GROSS should submit to both grid and local resources.

## 4.2 CMS Data Challenge 2004

The main components of the challenge were:

- Event reconstruction at the Tier-0 at a rate of 25Hz for 1 month;
- Transfer of the RAW and reconstructed data to Tier-1 and Tier-2 sites; and
- Analysis at remote sites.

Prior to DC04 70 million Monte Carlo events had been prepared and moved to the CERN Tier-0. These events were fed to the reconstruction farm, comprising ∼500 CPUs, at a rate of 40 MB/s. The reconstructed output from this farm, at a data rate of ∼4 MB/s, was stored at the Tier-0. Both the RAW and RECO data were then copied from the Tier-0 to remote sites where analysis jobs were run.

At the time of DC04 most of the components listed in the computing model did not exist. It was during this time that the forerunners to these systems were developed.

### 4.2.1 CMS analysis framework

The forerunner to the current CMS analysis framework was called ORCA (Object-oriented Reconstruction for CMS Analysis) [47]. The ORCA framework was based on POOL but was limited to accessing local data, which restricted users to a small subset of the available computational and storage resources.

### 4.2.2 Event data model

The event data model used during DC04 was significantly different to that described in the computing model. RAW data was split into two data tiers called Hits and “Digis”, following the GEANT terminology [48]. The Hits, only available for Monte Carlo data, contained the simulated physics process and held the Monte Carlo truth while Digis held the digitised detector response. Most analyses, at the time, required both the detector response and Monte Carlo truth and so required both data tiers. A new label, the owner name, contained both the data tier and software versions used. When specifying data for analysis both the dataset and owner name were required. For the remainder of the chapter, unless otherwise stated, the term “dataset” refers to the unique dataset/owner pairing.

CMS data were split into two file types: event and metadata. The event data contained the actual events while the metadata indexed the collections held within the event files. When ORCA required a certain event within a dataset it would look in the metadata to find a pointer to the relevant file and event collection. When the specification for a Monte Carlo dataset was created, so was the metadata, and this was referred to as “virgin” metadata. Each run within the dataset was generated separately against the virgin metadata. Once all of the runs had been finished, the virgin metadata files were populated with information about all runs, a process which required access to all files. This “final” metadata knew about all event collections within the dataset and allowed an analysis to proceed.

Due to an unfortunate naming convention, datasets with the same owner name, i.e. data tier and software version, all had metadata files with the same name. This, and the need for simultaneous access to all event data to populate the metadata, complicated their registration to the RLS. As a temporary measure during DC04 the virgin metadata files were placed into an archive that was registered with the RLS. ORCA could run on the virgin metadata if the ORCA configuration file explicitly listed the event collections to be used.

### 4.2.3 CMS software available at sites

Due to the large size of ORCA and its dependencies (more than 600 MB) it was decided that each site would have a centrally-installed copy. A job needed only to carry its executable and private libraries with it and could rely on a pre-installed version of all standard ORCA libraries to be available. For each version of ORCA installed a tag was added to

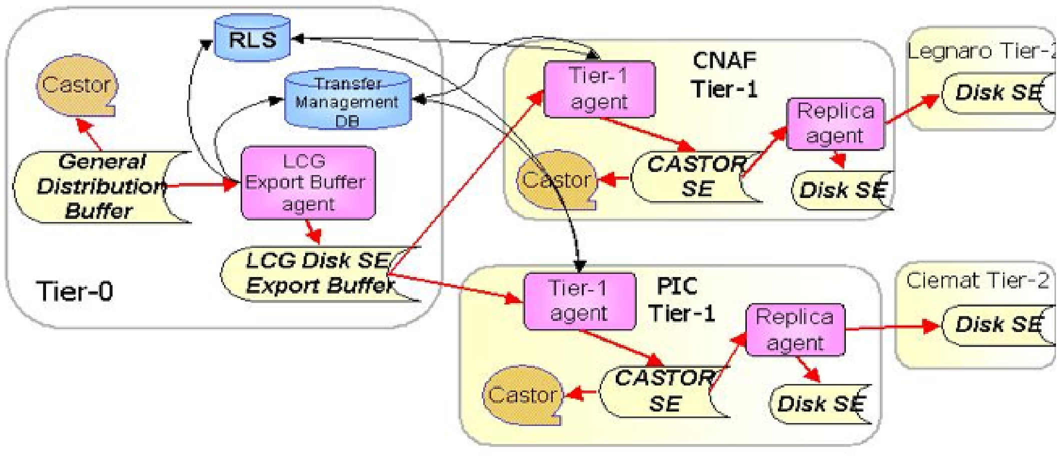


Figure 4.1: The DC04 data distribution system. [50]

the information published by the site in the grid information system. When a job was submitted this allowed the RB to locate all sites with the required software and thus steer the job to an appropriate location.

### 4.2.4 Data replication and discovery

Data movement was managed by a group of semi-autonomous software agents collaborating through the Transfer Management DataBase (TMDB). This is illustrated in Figure 4.1 [49, 50]. Data was copied from CERN to the sites using standard LCG replication tools. These data were stored on LCG SE's with at least two copies of each file, one located at CERN and one at a Tier-1 site. The files were registered with the RLS. During DC04 the RLS file metadata attributes were used to store information about the event collections contained within the file, these being dataset, owner and run number.

When preparing analysis jobs data discovery was handled through the RLS. Queries were performed with the POOL file catalogue commands. The list of LFN's returned were then listed in the analysis job JDL. Upon receipt of the job the RB looked at the list of LFN's and used the RLS to find sites that held replicas of the required files. The RB then submitted the job to a site within the list.

### 4.2.5 BOSS

CMS job tracking and submission at this time was handled by BOSS (Batch Object Submission System) [51]. BOSS was initially developed as a tool for MC production activities at CMS sites and provided a uniform interface to a variety of batch schedulers. It had no CMS specific concepts and thus required directing by a CMS specific application. For production this was McRunJob [52] but no similar tool existed for analysis. McRunJob created CMS production jobs and used BOSS for job submission and tracking. The advantages of this were that McRunJob could concentrate on CMS-specific functionality (in particular interfacing with the CMS MC production database, RefDB) with BOSS

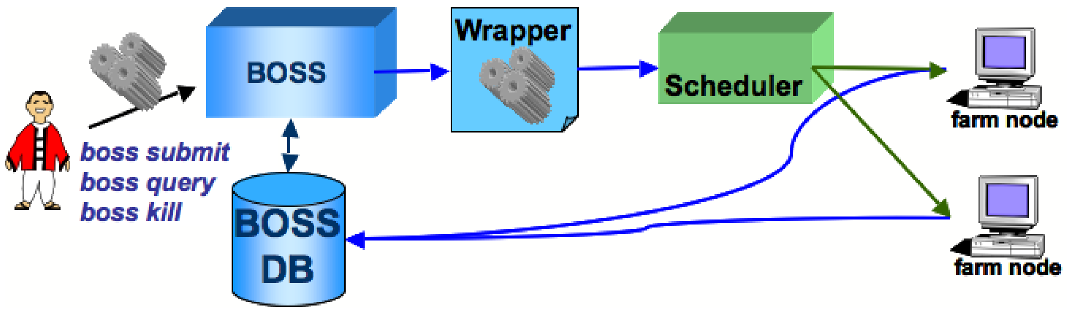


Figure 4.2: The basic BOSS workflow. [51]

responsible for all common job actions, such as submission and tracking.

BOSS was not a batch scheduler (the workflow is shown in Figure 4.2) and relied on an underlying implementation interfaced via a plugin mechanism. Each batch scheduler was registered to the BOSS system along with a variety of simple wrapper scripts designed to perform standard job actions (submit, query etc.). These were saved to the database with no changes to the core code. When a user requested that BOSS interact with a particular scheduler, the script for the desired action was retrieved and executed. This enabled BOSS to present a uniform interface to multiple schedulers.

For persistency BOSS used a MySQL database, which was used to store configuration options and job details. Persistent job information stored to the database included details given during job creation such as executable name and input and output files. Once submitted a job returned monitoring information directly to the database.

BOSS also allowed the user to specify the type of executable in the job, called "job type", to activate customised monitoring. By registering a job type a user could specify values that were monitored in the application's standard input, output and error streams. This monitoring was performed by a set of scripts, with one for each of the streams that required monitoring. To register a job type the user had to specify a schema that listed the variables to monitor and the set of scripts responsible for monitoring the streams and returning the correct values. For each job type a new table was created in the database with a separate column for each parameter. Each job was represented by a single row and only the last value in each category was retained. When a user specified the job type the relevant scripts were retrieved from the database and sent with the job. This customised monitoring had been used within CMS previously to, for example, record the current event number and allow the job's progress to be monitored.

User interaction with BOSS was via a Command Line Interface (CLI) or API. The API was a basic C++ API that accepted the same options as the CLI. Simple jobs could be defined by options to the CLI (or API), e.g.. executable name, type and any arguments, while a more complicated job could be described in a Job Description Language (JDL) file. The JDL file was written in the same Classified Advertisement (ClassAd) [31] syntax as the LCG JDL file.

Job creation and submission could be combined in a single step or split into two distinct operations. When a job was created BOSS processed the ClassAd file (if used), performed basic checks and saved it to the database. The job was then ready to be submitted. Upon job submission BOSS retrieved the relevant job information from the database and created various submission files. These consisted of files the user requested be sent with the job, the executable and various BOSS programs and files. The BOSS files sent included a job specification file, the monitoring scripts (optional), a program responsible for sending monitoring information back to the database (dbUpdator) and the main BOSS job wrapper (jobExecutor).

Once the job started the jobExecutor took responsibility for starting the monitoring, running the user's executable and performing cleanup actions. The jobExecutor wrote a log file (journal file) containing a complete log of the job's activity, including general details (start time, execution host, etc.) as well as the information provided by the monitoring scripts. The dbUpdator monitored the journal file and relayed any new information to the database.

By using the BOSS client the user could query the status of their job. BOSS would first look into the database and, if it was unable to determine the status from this (because for example the job had been submitted but no output had been retrieved) it would query the scheduler.

When the job finished the output was returned to the user. The output was either saved to a shared filesystem if submitted to a local scheduler or saved to the RB in the case of LCG submission. To obtain output in the LCG case a separate command had to be run. In both cases the output files included the output requested by the user as well as the journal file and the standard output and error. If the dbUpdator was unable to contact the database during execution (generally due to a firewall) then a command could be run that parsed the journal file and populated the database.

## 4.3 The GROSS design Process

Before work on GROSS began a full design process was completed. The first step was to determine what functionality and features were required from such a tool. Prior to starting this work an LCG report titled "Common use cases for a HEP common application layer for analysis" (HEPCALL II) [53] was published. This document looked at the various types of analyses that would be conducted at the LHC and what software would be needed to facilitate them. Differing scopes of analysis were studied ranging from end user analysis to large-scale managed analyses. Various levels of interactivity were also investigated ranging from fully interactive through to pure batch work.

The end user analyses described in the document included the non-organised analysis régime, where physicists perform un-coordinated work and require as much assistance as possible from the analysis system. The functionality for such a system included file access, job submission and robust provenance mechanisms. The starting point for such an analysis was a query requiring data that met certain criteria (both in terms of data

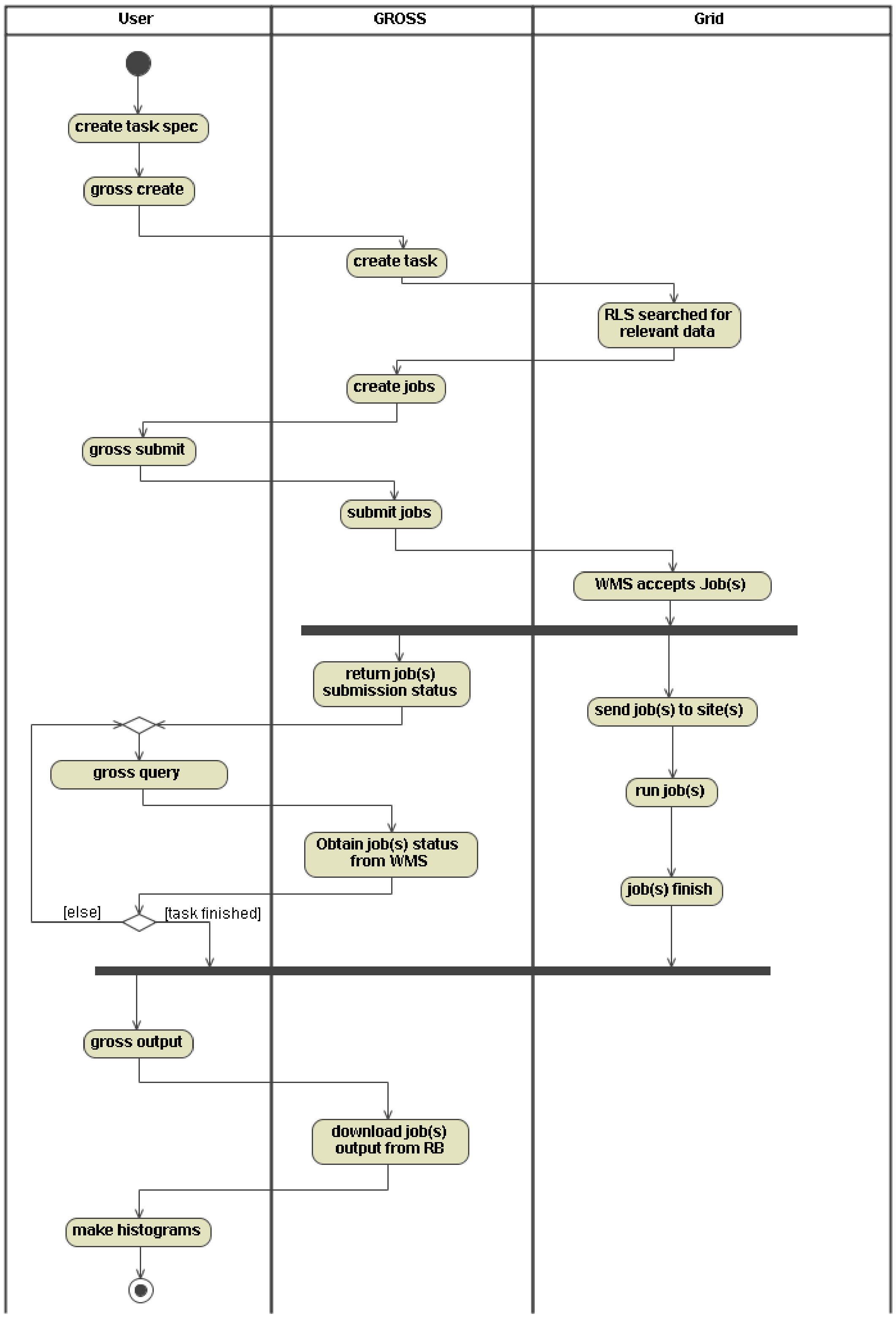


Figure 4.3: An activity diagram showing a GROSS workflow consisting of task creation, submission and output retrieval. The User swimlane shows the users actions and the GROSS commands used. The GROSS and Grid swimlanes show the resulting actions of both entities. Note that in this case no output is downloaded until all jobs have competed but this is not compulsory. Output from a job is available for download as soon as the job is finished.

quality and physics attributes), with the user not knowing where the data was located or how to access it. It was also possible that the user had modified parts of the experiments standard analysis code especially for their analysis. The need for logging and provenance information was emphasised with the example of a user attempting to determine where and when certain jobs ran in order to verify their results.

The HEPCAL II document included various models of analyses with varying levels of involvement by the grid WMS in operations such as data discovery and task splitting. The LCG/EGGE did not provide such high levels of functionality and together with the requirement for GROSS to be able to use non-grid resources resulted in GROSS being developed along the lines of the "No special support by WMS" model. This resulted in GROSS performing all data discovery, task splitting and job preparation. The WMS treated jobs as simple independent jobs with only a list of site and input data requirements, all features provided by the LCG.

BOSS was a named example of an application with an intermediate level of interactivity where the user had no direct control but did have the ability to monitor closely the process. As GROSS was designed to be used with fully developed analyses interactive control was not required.

From the HEPCAL II description of non-interactive end user analysis with an analysis system designed to work with no special support from the WMS a GROSS use case document was produced. GROSS user requirements were then derived from this document.

The main use case developed for GROSS is illustrated in Figure 4.3. This workflow described an end user wishing to submit an analysis task and obtain output files. The user had to provide information including ORCA executable, version, configuration file, output file names and the physics data selection query. GROSS would then contact the physics catalogue to discover the list of matching datasets and files. From this GROSS would optionally split the task into multiple jobs, each accessing a subset of the requested data. The task and its sub-jobs would be saved and a numeric identifier (id) returned to the user. With this id the user could then submit the task. Each job was submitted with full monitoring and tracking. At the end of a job the output files would be copied to an SE or brought back with the output sandbox. By running a separate command the user could retrieve the output sandboxes for all jobs in a task.

By the time of DC04 it had become clear that there would be no physics metadata catalogue capable of accepting physics queries and returning a list of matching data. GROSS therefore required the user to specify the requested dataset name when creating the task. If, at a later time, such a physics database was implemented the structure of GROSS would easily allow its use.

Existing tools were utilised whenever possible, minimising duplication of effort and reducing development time. BOSS was used for job tracking, monitoring and submission. Basing GROSS on BOSS allowed GROSS to take advantage of the extensive bookkeeping and job submission features provided within BOSS. GROSS provided the user interface and functionality specific to CMS analysis.

## 4.4 Architecture

One of the main design requirements for GROSS was that it be as modular and flexible as possible. Figure 4.4 shows the main GROSS functions and how they relate with external entities. Internally the different components interacted through well-defined interfaces that isolated the rest of GROSS from any component changes. During development the POOL file catalogue API underwent a number revisions but the structure of GROSS restricted the impact of these changes to one component.

GROSS was written in fully object-oriented C++ and utilised design patterns where appropriate. Design patterns are general repeatable solutions to commonly occurring problems within computer science [54].

Despite designing GROSS with the primary purpose of creating and submitting ORCA analysis jobs it was decided that GROSS should also be capable of submitting any type of application. Any application-specific functionality had to be part of a modular system. This allowed the future possibility of handling different types of CMS applications, e.g. Monte Carlo production and fast simulation. This was achieved by use of the "Abstract Factory" design pattern [54], which provides a mechanism for instantiating a family of classes in a uniform and consistent manner. A different family was needed for each supported application (called task types by GROSS). The family provided functionality suited to the particular application including task splitting and submission file preparation.

As BOSS masked the differences between schedulers it had been the original aim that GROSS would remain independent of any scheduler choice, leaving the user free to choose at the time of task submission. However, the large differences in both the execution environment and data access model between grid and non-grid resources made complicated this. A different task type therefore had to be used for ORCA jobs submitted to the grid compared to jobs submitted to a local scheduler. These differences may have been hidden by a sufficiently high abstraction, however this would result in such widely varying requirements for the two cases so as to make it impractical. It was decided that the benefit to the user was outweighed by the development work required, as generally users know if they are submitting to local or remote resources.

The family responsible for ORCA grid jobs is shown in Figure 4.5. On the left is the abstract factory that is responsible for instantiating the required classes. When creating a task the type was specified on the command line and the relevant factory implementation instantiated all classes in the family. The task type was saved to the database along with the task so that correct classes could be correctly instantiated in further operations. The family was composed of classes responsible for the task, the jobs, wrapper steering file and JDL. There were two concrete classes responsible for tasks and jobs, one responsible for creating the object from the users specification and the other for retrieving information from the database.

As GROSS was closely coupled with BOSS it was decided to distribute them together. A user only needed one package and both were installed and configured simultaneously.

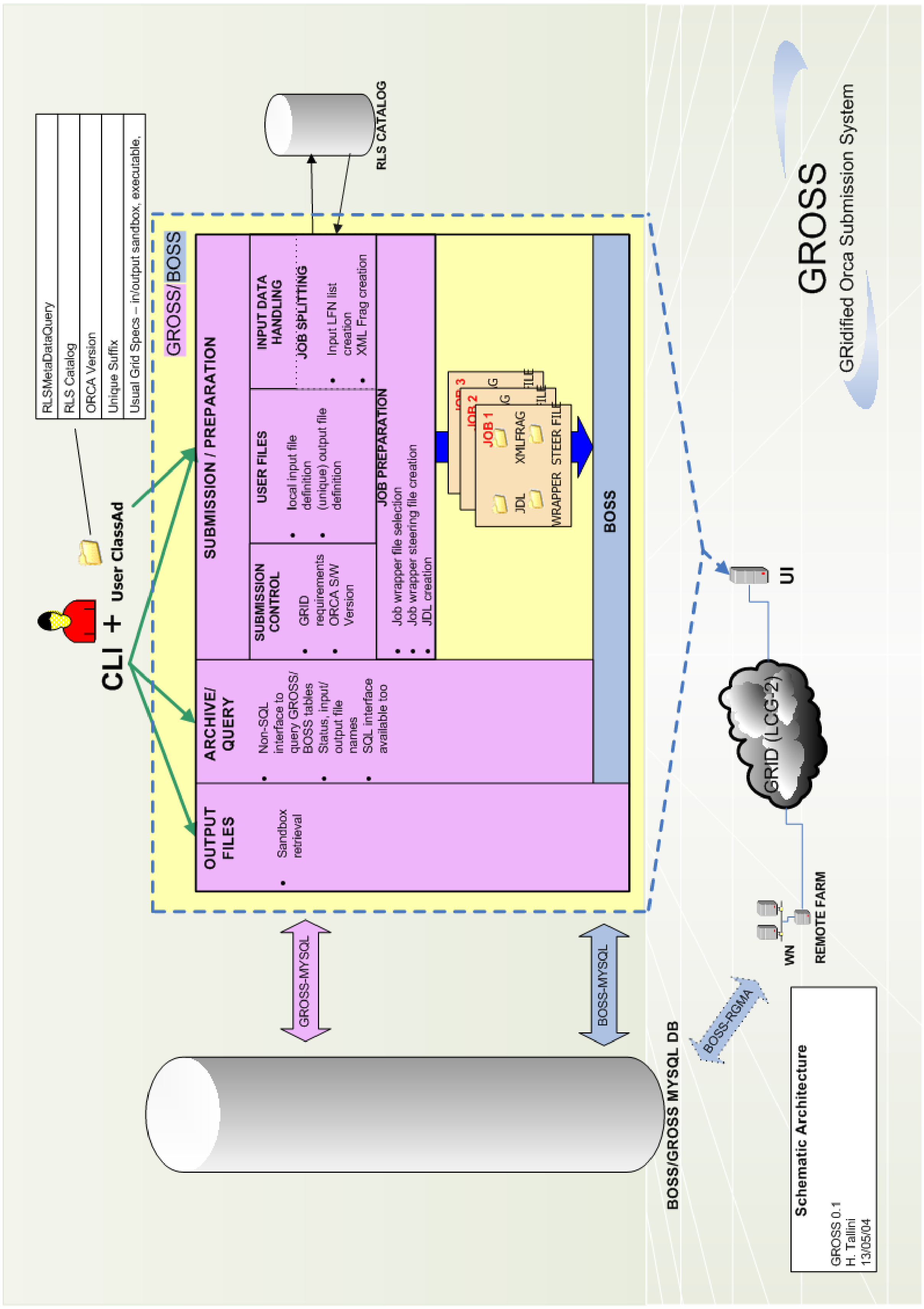
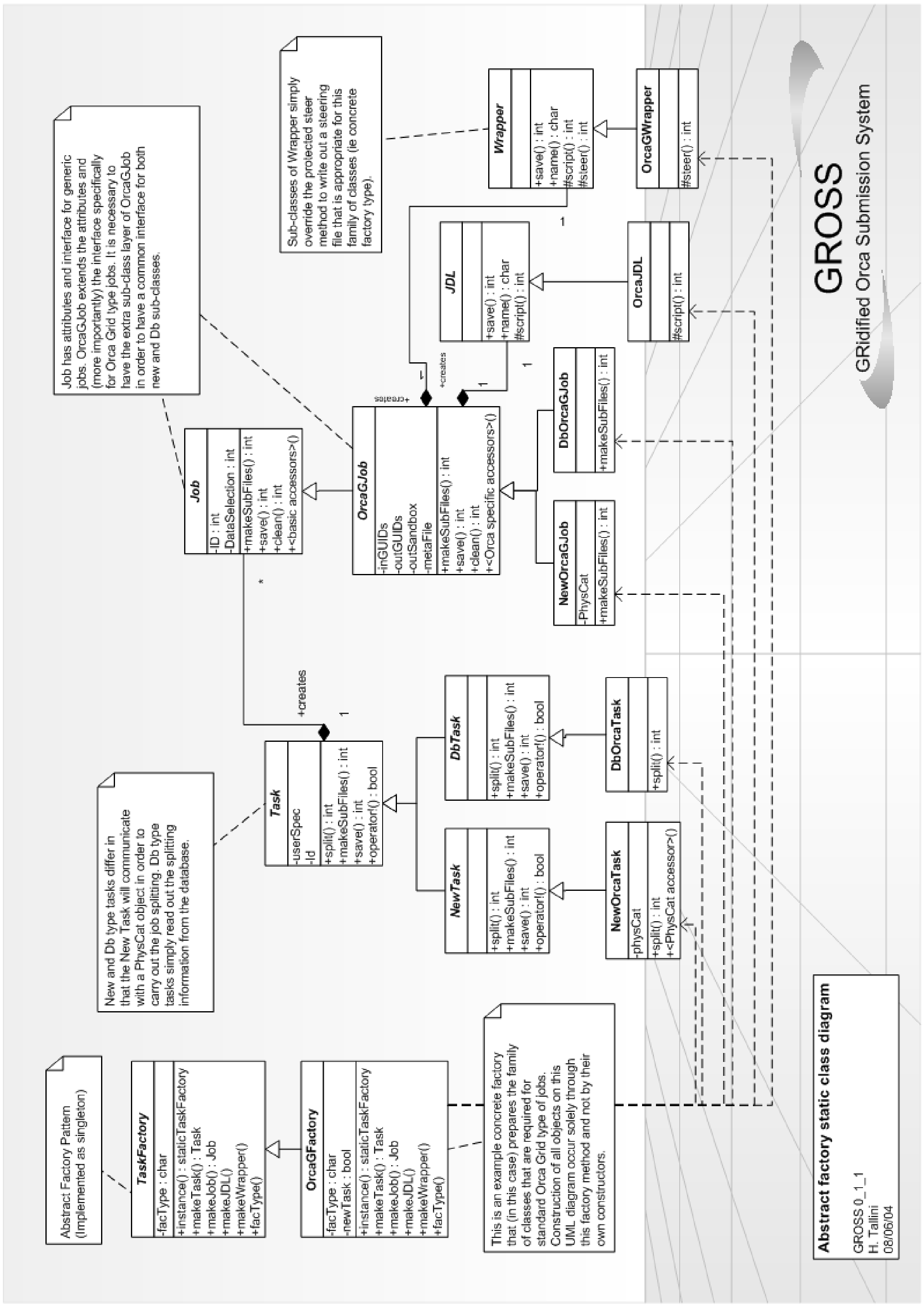


Figure 4.4: The main features of GROSS and their relation with external entities. [1]


Job has attributes and interface for generic jobs. OrcaGJob extends the attributes and (more importantly) the interface specifically for Orca Grid type jobs. It is necessary to have the extra sub-class layer of OrcaGJob in order to have a common interface for both new and Db sub-classes.

Sub-classes of Wrapper simply override the protected steer method to write out a steering file that is appropriate for this family of classes (ie concrete factory type).

New and Db type tasks differ in that the New Task will communicate with a PhysCat object in order to carry out the job splitting. Db type tasks simply read out the splitting information from the database.

This is an example concrete factory that (in this case) prepares the family of classes that are required for standard Orca Grid type of jobs. Construction of all objects on this UML diagram occur solely through this factory method and not by their own constructors.

Abstract Factory Pattern (Implemented as singleton)

Figure 4.5: Class diagram showing the abstract factory. "OrcaG" refers to the classes responsible for ORCA tasks that are submitted to LCG. [55]

As in BOSS it was decided to store all GROSS task information into a database. As most job information was stored into the BOSS database only a few extra tables were required. Thus it was decided to add the GROSS tables to the BOSS database. GROSS access to this database was handled through a singleton [54]. A singleton is a class that can only be instantiated once, resulting in only one database connection for the whole session. Database calls from all GROSS classes were routed through this object to minimise the database load.

## 4.5 Functionality and interface

The user interacted with GROSS through a command line client. Options were passed on the command line with the task specification passed via a file (the task specification file). As both BOSS and LCG used ClassAds for their configuration information it was decided to utilise these in GROSS. This allowed GROSS to take advantage of a mature parser library and gave the user a familiar configuration syntax. An added advantage was that unknown fields in the configuration file could be passed down to BOSS, where they might be understood, or, if not, down further to the scheduler. This provided a mechanism for users to provide configuration options direct to the LCG scheduler.

### 4.5.1 Task Creation

Task creation required command line options including task type and the specification file. This file listed all information needed to create the task including the executable file, paths to any libraries required by the application, ORCA version, ORCA configuration file path, the dataset name and output file name. As well as a unique id each task had the option of having a user-defined name associated with it. This was appended to output filenames to aid recognition.

### 4.5.2 Input data handling

At the time of DC04 the expected minimum necessary information required for data discovery was a dataset name. From this GROSS had to be able to discover all data files in the dataset. To allow refinement of this, it was also possible for a user to reduce the scope of the analysis with a further query on the RLS metadata, i.e. on the run number.

The ability of the POOL file catalogue to use multiple backend technologies allowed a user to switch between the official CMS RLS and a private POOL file catalogue by changing the file catalogue contact string in the task specification file.

POOL had many dependencies including SEAL [56] (another LCG project), XML (if used with XML catalogues), MySQL (if used with MySQL catalogues) and BOOST [57] (a set of C++ libraries). GROSS therefore shared these dependencies. These dependancies would normally have been unacceptable but it was assumed that ORCA, which also required POOL, would be installed to allow the user to develop their analysis.

Due to the predicted load from the whole collaboration it was decided to reduce the number of queries to the RLS originating from GROSS as much as possible. To accomplish this GROSS implemented a local cache of the relevant information in the RLS. The POOL API provided functionality to extract information from one catalogue and store it in another. Thus the first file catalogue operation performed by GROSS queried the RLS for all information belonging to a dataset and saved this information to a local XML catalogue. All further file catalogue queries were performed on this local copy that, as well as reducing load on the RLS, also improved performance.

One major limitation of the POOL file catalogue API was a lack of “OR” logic in query terms; query terms could only be joined with “AND” logic or wildcards. GROSS required this functionality for the users optional metadata query, which typically involved queries containing a list of run numbers. To implement this the GROSS POOL interface contained functionality that parsed all query strings for the “OR” token. If found the query was split at this point. Each query was then executed separately with the final result set formed from the sum of all partial result sets. Thus, from the user’s perspective, “OR” functionality was natively supported.

### 4.5.3 Task splitting

Each CMS Monte Carlo production job generated a separate run of data, hence it seemed logical to split the analysis task in the same way. This allowed maximum flexibility for job submission. Each job required the minimum number of files for a given number of events and each file was needed by only one job, so allowing each job within a task to be sent to a different site. This splitting was only appropriate for an analysis case where the majority of events were fully analysed, as was the case with most CMS analyses in 2004. This would not be the case with real data where not every event in a dataset would prove interesting to every analysis. Hence this strategy was appropriate for Monte Carlo analyses but not for real data.

The task splitting workflow is illustrated in Figure 4.6. Once the local file catalogue containing all files within the dataset had been created, a list of runs was produced subject to the optional user-provided metadata query. This list was then used to create the sub-jobs. Once jobs had been configured with their run number they could then obtain a list of their input data files by querying for all of the LFN’s for their run. The archive file containing the metadata also had to be discovered and added to the list of input files for each job.

Each job was configured with the correct data selection, input and output data files. Files generated during configuration included a POOL XML fragment listing the input data, a JDL file for the RB and a configuration file for the wrapper script that ran the appropriate workflow on the worker node. All information was saved to the database, so in subsequent actions the entire task and all jobs could be recreated without repeating the data discovery or task splitting.

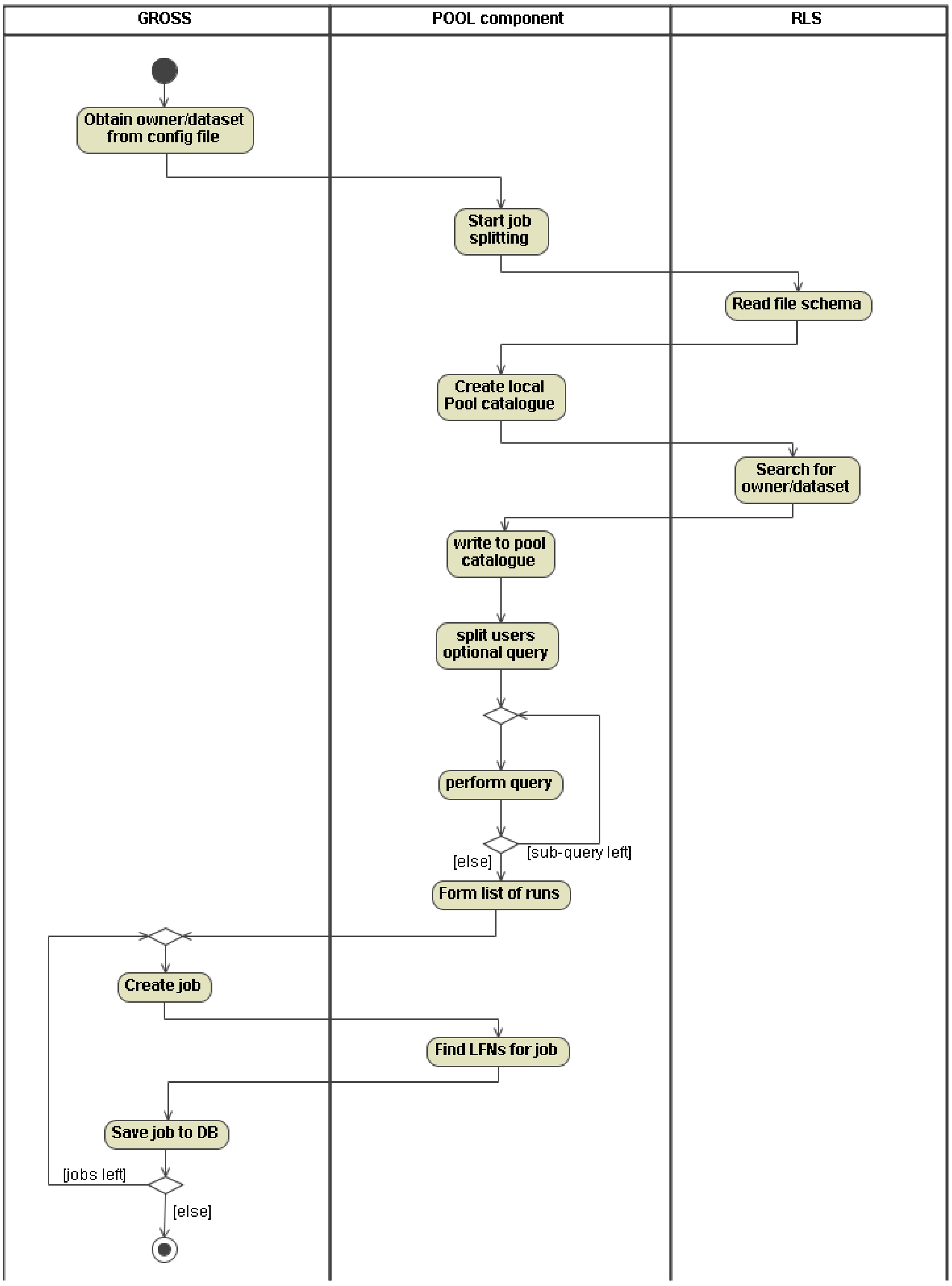


Figure 4.6: Activity diagram showing the task splitting workflow.

### 4.5.4 Task submission

To submit a task the user provided the task id and the desired scheduler. GROSS then retrieved the task and sub-jobs and prepared the submission files. These files included the user-defined files, ORCA configuration file, user's executable, user libraries, wrapper script and the wrapper script steering file. Each job was submitted to BOSS with the GROSS job type to activate the specialised monitoring. After submission each job record in the GROSS database was updated to point to the BOSS job record, which contained information such as scheduler id and status.

### 4.5.5 ORCA wrapper script

Before ORCA could be run a number of steps had to be taken: the CMS environment needed to be set, input files located, the user's executable and libraries prepared and many other operations. In order to perform these tasks a wrapper script was necessary. The script had to be lightweight, fast and capable of running on many different types of operating system and architecture. This script was implemented as a shell script that provided a capable but lightweight solution.

Once a job began on a worker node a process similar to the job preparation step, but in reverse, was run. The local scheduler system started the BOSS jobExecutor, which had to set its own environment before starting the GROSS wrapper. The GROSS wrapper script then set the local CMS environment before starting ORCA.

The wrapper script obtained its configuration information from the steering file. If the script was unable to locate this file the wrapper aborted. Once located this file was parsed and its contents (key-value pairs) placed into variables within the script. After this a check was performed to confirm that certain critical variables were defined. These variables included executable name, ORCA version etc.

The CMS software area was located using an LCG-defined environment variable. In this area was a script, created by the CMS software installation, that set up the CMS environment. After the CMS environment was available the ORCA environment was required. The GROSS wrapper script took the version of ORCA requested, checked its availability and sourced its environment.

Next the input data files had to be located. It was LCG policy to send jobs only to sites that hosted all of the required files on their SE. The LCG SE supported three file access protocols: direct file access, RFIO and GSIFTP. Direct file access assumed that files were available from the worker node filesystem. This mechanism was deprecated due to scalability issues and was dropped from later LCG releases. RFIO was a widely supported file access mechanism within HEP that allowed files stored on a server to be accessed from remote clients and was the recommended access method. It required that applications support the protocol, which most HEP applications at the time did. The GSIFTP protocol was the standard mechanism for transferring files between LCG SE's. It was an FTP implementation that utilised the LCG security model (X509 certificates).

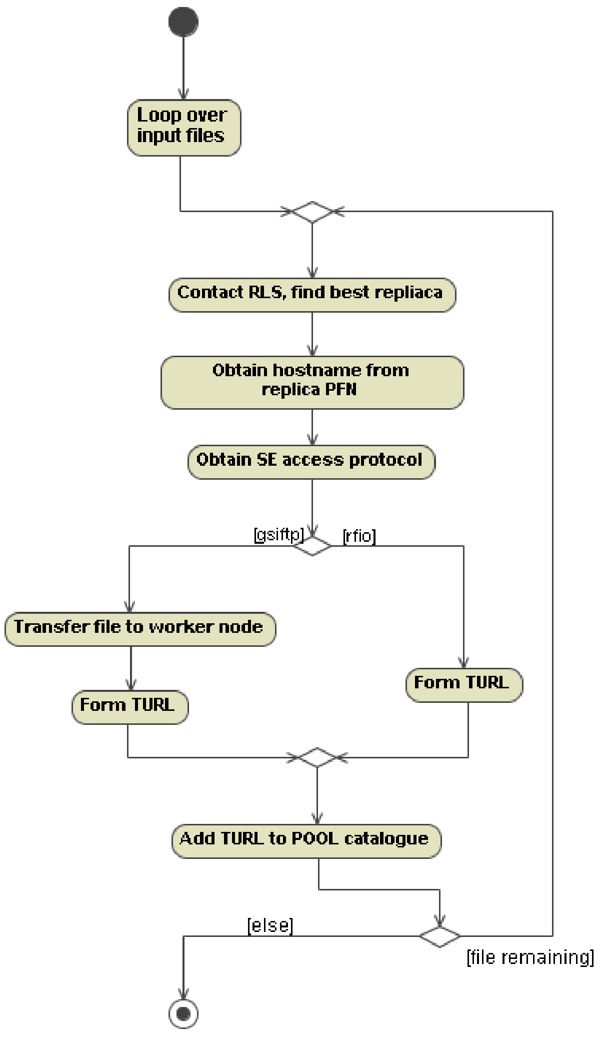

Loop over input files
Contact RLS, find best repliaca
Obtain hostname from replica PFN
Obtain SE access protocol
[gsiftp]
[rfio]
Transfer file to worker node
Form TURL
Form TURL
Add TURL to POOL catalogue
[else]
[file remaining]

Figure 4.7: Activity diagram showing wrapper script workflow for locating input files. The deprecated direct file access protocol is not shown.

ORCA could not use GSIFTP for file access and so files first had to be downloaded to the local filesystem. This resulted in the use of unnecessary bandwidth between the SE and WN and the need for temporary disk space on the worker node. By definition an LCG SE had to support the GSIFTP protocol whereas the others were optional. This meant that the script had to discover the access protocols provided by the SE and take appropriate action.

A site may have had more than one SE hence, for each file, the hosting SE had to be found and the supported protocols discovered. The workflow is illustrated in Figure 4.7. This information could be obtained from the brokerInfo file [30], a file created by the RB on job matching and sent with the job, containing information about each SE at the site that hosted input data, including supported protocols. Thus for each file the script obtained the location of the closest replica from the RLS including the hosting SE. Using the information in the brokerInfo file the SE's list of supported access protocols was discovered. The scripts then applied rules to find the correct access mechanisms, trying

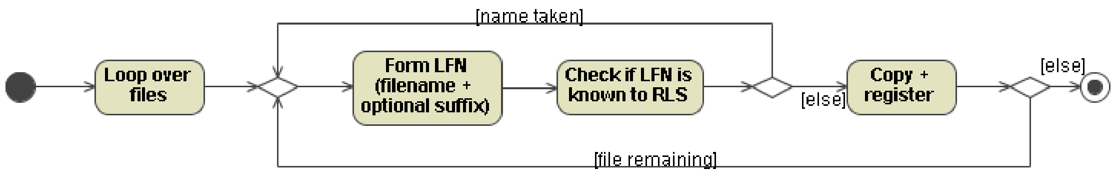


Figure 4.8: Activity diagram showing wrapper script workflow for registering output files with the RLS. If an LFN already existed GROSS did not overwrite it. It was the user's responsibility to ensure that output from previous tasks with the same optional suffix no longer existed.

in the order RFIO, file and then GSIFTP. Once the access protocol had been discovered the script had to form a correct contact string, possibly copy the file locally (in the case of GSIFTP) and update the XML POOL fragment, sent with the job, with the new contact string.

Even if the metadata archive was available over RFIO it was copied to the local disk to allow the metadata files to be extracted. The metadata archive contained several POOL XML fragments describing the metadata files. Once extracted the script merged these with the POOL fragment from GROSS and processed each of the files, setting the contact strings to point to the newly-extracted files. The metadata was virgin, therefore ORCA could not read it and go straight to an event within a data file. A program distributed with ORCA first had to be run on the event data files to find the event collection identifier, which was then added to the ORCA configuration file to allow ORCA to access the events.

Before running the user's executable a check was made to ensure that all of the required libraries were available. If libraries were missing the script exited with an error. ORCA was capable of providing useful output but still exiting with a non zero exit code, in this case the script continued but recorded the exit code for later use. Once an output file was found, it had the optional task name appended and was either saved to an SE and registered with the RLS or returned via the LCG output sandbox. Figure 4.8 shows the workflow for registering output files with the RLS. Files were saved to an LCG SE local to the the site and registered with the same LFN as their filename.

Finally the script removed all files it had created and exited with the same error code as that of the user's ORCA executable. BOSS recorded the exit code (and various statistics), collected all of the output files and exited.

### 4.5.6 Querying and output retrieval

Once the task had been submitted the status of its constituent job's could be queried. If GROSS was unsure of a jobs status, i.e. the job had been submitted but not retrieved, GROSS queried BOSS. BOSS then queried the database for the job state, and, if this could not be determined, the scheduler was queried.

All GROSS jobs were submitted to BOSS as a member of the GROSS job type. This job type was registered during the GROSS installation and registered monitoring scripts designed to monitor the wrapper error and information messages. If the running job was able to contact the GROSS database then these messages were continually relayed to the database. The database held the last message in each category for each job. This information was accessible from the GROSS command line.

Once a job had finished the output could be retrieved. When a user requested that a task's output be retrieved, GROSS located the jobs that had completed and asked BOSS to retrieve their output files. If the output was retrieved successfully the GROSS database was updated. By using the GROSS client the user could later determine where a particular task's output had been saved.

### 4.5.7 Local farm submission

Job preparation and execution on a local batch farm required significantly different steps from a grid submission, requiring the introduction of another task type. This task type assumed that the user had a POOL file catalogue containing a locally available dataset. This may have been the RLS in the case where a user wished to bypass the LCG and submit directly to local resources. This only worked in the case where the registered PFNs were also the file contact string, which was the case at CERN but not at LCG sites in general. The difference in data discovery between the two task types was that the local type resolved file contact strings at creation time whereas grid jobs determined this at runtime given the site at which they ran.

The local version of the wrapper script provided similar functionality to the grid script. File access was simplified as it was assumed that the POOL fragment already had correct contact strings. Not all local batch systems supported input or output file handling, therefore all files had to be available on a shared filesystem. At the end of a job the output was copied back to another location on the shared filesystem specified when the task was created. The GROSS output retrieval step was not necessary.

## 4.6 Performance and useability

Unfortunately GROSS development progressed more slowly than originally envisaged and DC04 had already begun by the time it was released. Due to the amount of data to be analysed an automated procedure using custom software agents had been developed (Figure 4.9). These agents were tightly coupled with the transfer system. Once files appeared at CERN and were registered in the RLS they were transferred to a remote site, the replica registered and an analysis job sent via LCG. The executables and libraries were sent together with the job and the output was saved to an SE and registered to the RLS. Job submission and monitoring were provided by BOSS. In the last two weeks of DC04 15,000 jobs were sent, with a grid efficiency (i.e. the efficiency for a job to be executed at a site irrespective of the success of the application) of 90-95% and a latency of 20 minutes between files appearing at CERN and their remote analysis [58, 50].

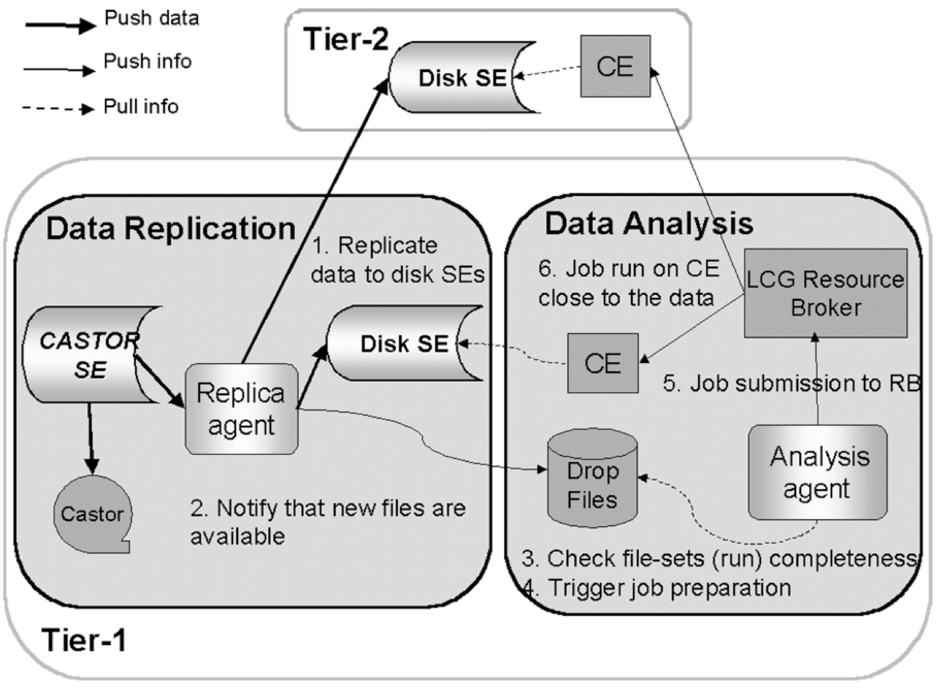


Figure 4.9: DC04 real time analysis chain. [58]

During this time the developers evaluated, tested and made several enhancements and bug fixes to GROSS. While all functionality worked as intended the performance was worse than expected. Specifically the data discovery step was found to be unacceptably slow, which was traced to poor performance of the RLS.

The RLS consisted of several components: the Local Replica Catalog (LRC), which maintained the list of replicas at a site, the Replica Location Index (RLI), which indexed multiple LRC's and the Replica Metadata Catalog (RMC), which held file metadata [59]. Within LCG the RLS had only been implemented as a single LRC (and RMC) per experiment instead of per site as it had been designed. The LRC had been evaluated with inserts, deletes and queries on LFN's. These tests and experience during DC04 showed acceptable performance [60, 61]. However, metadata tests had only involved inserts and deletes. Once the system was queried for all files meeting certain criteria it became apparent that performance was unacceptable [60, 61]. Queries for all files belonging to an average CMS dataset ( ∼1000 runs), as required by GROSS, took 2–3 hours to complete [60].

The real time analysis was only concerned with LFNs and so was able to avoid the poor performance of the RLS with metadata queries. GROSS, however, was based on the premise that the user did not know the files they wished to analyse, only the dataset. One way to improve performance was to use the POOL file catalog commands manually to perform the search of the RLS and to store the results in an XML catalogue. If this was then used with GROSS and multiple tasks were created against the dataset, performance was seen to improve with data discovery taking minutes rather than hours.

## 4.7 Post DC04 distributed analysis

After DC04 CMS decided that instead of requiring LCG to optimise the RLS it would eliminate the need for a global file catalogue. Instead a central catalogue would keep a mapping between datasets and sites. Each site would then host its own file catalogue (possibly more than one). In this model the central catalogue would contain significantly less information and be subject to fewer simpler queries, which improved performance. Each site had a web-accessible service, known as a PubDB, which would return the correct file catalogue(s) for a dataset [49]. The file catalogue could be any POOL file catalogue and was generally either XML files or a MySQL database.

The data movement system was re-engineered but still consisted of transfer agents at each site controlled by a central database. The site agents periodically queried the database and discovered any outstanding data requests. The agents downloaded the relevant files and marked the transfers as complete in the database. Only event data was known to the CMS data management system, hence, before the dataset was available for analysis the metadata had to be downloaded in a separate publishing step. Here the virgin metadata was downloaded from CERN and populated. These and the event data files were then added to the local POOL file catalogue(s).

The result of these changes was that jobs no longer needed to carry their own file catalogue or know *apriori* which files were required. It was the responsibility of the site to provide the job with the means to locate all files needed at runtime.

In the transition period between the phasing out of the RLS and adoption of the PubDB system an interim solution was used. Full (i.e. non virgin) metadata was made available via the CMS MC production system. Once a dataset had been produced it was copied to CERN, where the metadata was fully populated with information about all event data. These metadata files were then made available from the CMS web server. During the data discovery stage GROSS contacted both the RLS for event files and the CMS webserver for the metadata and a POOL catalogue fragment. These were then both sent with the jobs. This temporary procedure was only needed while the PubDB system was being developed.

## 4.8 CRAB

Around this time another tool called CRAB (CMS Remote Analysis Builder) [40], developed by a group from Italy, was released. This tool was similar to GROSS with only minor differences. During December 2004 a meeting was held in Bologna to decide the future of CMS distributed analysis. At this meeting a comparison of the two applications was carried out.

CRAB was written in Python and did not utilise BOSS. These choices and the relatively large development team allowed CRAB to develop PubDB support before GROSS. The lack of BOSS integration simplified the installation of the software and eliminated the requirement for a MySQL database but resulted in poor job tracking and logging. There was no central repository where information about all tasks could be found. Each task had its own directory for submission and output files and this directory had to be retained if the user wished to retain knowledge of the task.

CRAB also had other features aimed at simplifying its use. It was integrated with the CMS build system and thus, with the appropriate ORCA environment set, could locate the users executable and all libraries. GROSS, in contrast, required the full path to each to be listed in the configuration file. With CRAB, tasks could be created, submitted and scheduled for automatic file retrieval with a single command. The same workflow in GROSS required at least three separate commands. However, GROSS had a much clearer and more consistent command line.

Overall it was felt that GROSS provided a more uniform interface and better job monitoring and bookkeeping but CRAB was easier to install and provided many features that simplified the analysis process for the user.

As a result of the meeting it was decided that both applications would continue to be developed whilst an evaluation by the CMS physics community was conducted. CRAB would investigate using BOSS to provide job submission and logging and GROSS would adopt the PubDB system and some of the ideas in CRAB aimed at simplifying and automating the analysis process for the user.

## 4.9 New architecture and features

Until the meeting in Bologna GROSS had been distributed together with BOSS. Many sites, however, already had an installation of BOSS from CMS MC production activities, so it was decided to package GROSS separately.

The change to using the PubDB system required a large change in the architecture of GROSS. Previously each job had known about all input and output files, but this information was no longer necessary. GROSS now relied on a correct POOL file catalogue at the site to contain the requested data. The facility for users to send jobs using their own POOL file catalogue to local resources was still maintained.

Previously, analysis with GROSS had only focused on the Hits data tier, whereas most advanced Monte Carlo analyses required access to both the Hits and the Digis. This required access to data from two owner/datasets and possibly the pile-up dataset also. The PubDB system had been designed to cope with this and for each owner/dataset the central PubDB listed the related owner/datasets. In the task specification file the user could request data tiers related to the the specified owner/dataset. GROSS would then ask the central PubDB catalogue for their entries as well. Sites were not required to host entire datasets, only a continuous range of complete data runs. For each job GROSS required that a site hold the same run in all data tiers, plus the complete pile-up dataset. The ability to access multiple data tiers was only required for analysis of Monte Carlo data.

The move to PubDB improved performance considerably. The data discovery step was now simplified, when the task was created the only data discovery steps needed were to ensure that the dataset existed and that at least one working site had a copy. This was in contrast to the previous situation where every file within the dataset has to be located. The combination of the new system and simpler queries resulted in a considerable performance gain, with data discovery taking approximately 10 s.

During the interval between task creation and submission the status of the sites may have changed, so the list of sites was formed at submission time. GROSS contacted each hosting site to find the contact strings for the file catalogues. If a site did not respond within a set time it was skipped and GROSS moved to the next site. The list of appropriate sites was added to the JDL to allow the RB to send the job to the correct site and the list of file catalogues was added to the steering file for the wrapper script.

Previously GROSS had split a task into one job per run. However, as the time required for different analyses varied greatly this was modified so that the user could change this. As well as specifying the number of runs to analyse in a job the user could also specify the start and end run numbers if they did not want to analyse all runs within the dataset.

During Autumn 2004 LCG sites began to migrate their operating systems from RedHat Linux 7 to systems compatible with Scientific Linux CERN 3, SLC3 [62] (a RedHat Enterprise 3 derivative [63]). Applications built on SLC3 could not run on RedHat 7 systems. LCG sites specified their operating system in the information system. If a user wished to run on a certain architecture they could add a ClassAd requirement to the task specification file, which would then be passed to the RB at submission time. To allow access to all sites GROSS provided the option for users to send their source code and have their analysis application built on the worker node. If this was requested the user's code would be built in an ORCA environment on the worker node, with the resulting executable automatically linking with the standard ORCA libraries already present at the site. If any errors were encountered during compilation the job would abort.

During one of the LCG upgrades, limits on the input and output sandbox sizes were introduced. The RB stored these and a number of users were sending and/or retrieving very large files via them. This had the effect of filling up the RB's disk and affecting its performance. The limit imposed on input sandboxes was 10MB. GROSS sent the user's pre-compiled binaries as well as files needed by both GROSS and BOSS. The ClassAd library, which was sent to allow BOSS to read its configuration file, was about 7MB, which, together with the user's executable, easily exceeded this limit. It would have been possible to install the ClassAd library at each site, thus circumventing the restriction. However this would require support for each site along with a versioning mechanism to handle different BOSS versions. Thus it was decided that the BOSS developers should attempt to reduce the size of the libraries required by BOSS. A simpler version of ClassAds, called ClassAdLite, was developed which was only a few hundred kB.

In order to help reduce the output sandbox size the facility to allow files to be saved to a set path on a SE of the user's choice, without being registered with the RLS, was introduced. This was easier for users to use than the RLS as they could see files accumulate on an SE, e.g. their private area on CASTOR at CERN, rather than having to query a remote service.

Originally it was foreseen that automatic output retrieval would be handled by a cron-like entity calling GROSS at regular intervals to download any remaining output files. However, the availability of cron was not guaranteed at every site from which users submitted CMS jobs. An alternative method was chosen that made use of an agent process to periodically run GROSS. If, when the user submitted a task, they requested this functionality the script was started on the UI after submission. This script slept for a user-configurable time, then woke and ran GROSS telling it to retrieve any available output for the task. If no output was available the script would go back to sleep. When GROSS reported that all output had been collected the script would exit and if the script was still active after a user-configurable time limit (by default 2 days) it would exit anyway. This was to prevent it from running forever when a problem prevented retrieval of the output. The script wrote to a log file so that users could follow its progress and see how much output remained. If the user ended the session on that machine the process would continue, as it ignored SIGHUP signals. This feature was not enabled by default due to concerns about users starting large numbers of processes on popular machines.

Output files were stored in a separate directory for each job under the user-specified area, with the optional user suffix appended to each file. As a result of the task splitting it was not uncommon for a user to have hundreds of small ROOT files in different directories. This made keeping track of files or running ROOT over the files inconvenient. Therefore new functionality was introduced that allowed the user to obtain one root file for a whole task.

By running a script in the GROSS distribution the user could merge all the ROOT files within a task. It was also possible for the user to merge only the output from a subset of the jobs in the task. This script ran GROSS to discover the location of the output files, and any jobs that had finished but whose output had not been retrieved had their output collected. A search for ROOT files was then carried out and any found were added to the list of files to be merged. ROOT was then run, merging all the input files to create a new ROOT file in the specified location.

With jobs submitted to the LCG it was not uncommon for a number in any given task to suffer transient errors, either within the LCG workload management system or because of the configuration of an individual site. To help with this problem a mechanism for resubmission of failed jobs was added. Using this command GROSS would resubmit any failed jobs in a task, or job range within a task.

Real time monitoring was improved by modifying the GROSS job type to recognise the standard ORCA event number printout. This allowed users to monitor the current progress of their jobs.

## 4.10 Community Evaluation

Once most of these changes had been made GROSS was made available to the CMS physics community. GROSS was evaluated by members, including the convenor, of the Higgs PRS (Physics Reconstruction and Selection) analysis group. GROSS was used to perform analysis for $\tau$ jet studies and Higgs searches on Monte Carlo data.

The $\tau$ studies consisted of two areas of activity: jet calibration and HLT studies. The calibration study involved developing a Monte Carlo jet $\mathrm{E_T}$ calibration for $\tau$ jets. This was achieved by running over a sample of 100,000 hadronically decaying $\tau$'s and comparing the Monte Carlo and reconstructed jet $\mathrm{E_T}$. An algorithm was then developed in ROOT to relate the two. The HLT $\tau$ jet studies looked at trigger optimisation specifically for the MSSM heavy Higgs decaying to two $\tau$ jets.

Higgs analyses included the channels MSSM $H \rightarrow 2\tau$ jets, MSSM $H \rightarrow 2\tau \rightarrow \tau\mathrm{jet} + l$, MSSM $H \rightarrow 2\tau \rightarrow 2l$ and MSSM $H^+ \rightarrow \tau$. These analyses involved running over both signal and background datasets. Backgrounds included QCD multi-jets, W+jets and Z/$\gamma \rightarrow \tau\tau$. The analysis of signal events required the full analysis of the majority of events. Where the backgrounds were likely to be rejected they were preselected, during generation, to be more likely to pass the analysis cuts. Thus this testing used the majority of the available sample and hence did not represent a realistic indicator of performance with real data.

In these analyses 4 users created ∼220 tasks analysing 22 datasets. A total of 4,381 jobs were submitted to BOSS, of which 4,181 were successfully submitted to a scheduler. Of these 3,134 were submitted to LCG and 1,047 to resources local to CERN. 3,726 jobs finished successfully, representing a success rate of 89.1% and leaving 455 jobs that exited with errors. These errors generally indicated a problem with the user's code (possibly something that was not apparent during limited use) or a (transient) problem at a site, often with file access. It was not possible to give a detailed breakdown of the various failure modes as the ORCA exit codes did not always correspond to the actual error. Also, any user requested job resubmission would have overwritten the details of the previous job in the database. Thus the 89.1% success rate represented the rate allowing for re-tries. In total these jobs used over 605 days of computational time.

These statistics were taken from the GROSS/BOSS database. If a running job was unable to contact the database the records may be incomplete. In this case GROSS attempted to correct the situation at the output retrieval stage by running BOSS with the retrieved journal file. This process was not, however, always reliable.

## 4.11 Conclusion

GROSS met all of the requirements set out in the design phase and was a proof-of-principle tool for distributed analysis within HEP. It proved useful for physicists during work leading up to the writing of the CMS physics TDR [9]. It allowed a physicist to perform an analysis over large amounts of data distributed between CMS' resources worldwide. The use of GROSS by physicists in their work provided much needed-feedback and allowed the introduction of a large number of bug fixes and new features.

Compared to CRAB it was felt that GROSS provided better logging and monitoring, implemented through BOSS, but that CRAB was a more lightweight and flexible design. CMS decided that CRAB provided the more appropriate client-facing application, but recognised that GROSS was more robust and through BOSS provided many useful features. Thus it was decided to introduce many of the features GROSS provided, i.e. the task concept and handling, into BOSS, while CRAB would act as a CMS-specific application relying on BOSS to provide the task submission and logging functionality. The BOSS development work is described in the next chapter.

# Chapter 5

# Boss development

## 5.1 Introduction

BOSS had been used since 2002 within CMS production [64] and in DC04 with analysis [1, 40]. After DC04 it was decided that BOSS would be integrated into both systems to provie generic batch job functionality. The opportunity was taken to re-design BOSS completely to fit this role better, existing functionality was improved and new features introduced [2]. New features included the concept of a task and the possibility for a job to run more than one application. Improvements were made to the monitoring (removing the direct database connection) and logging (eliminating the requirement for a database server).

## 5.2 Limitations of previous versions

The lack of a task hierarchy required both CMS production and analysis systems to operate BOSS with an ensemble of individual jobs. A separate BOSS invocation was required for each job, with no sharing of information between instances.

The BOSS API did not offer any greater functionality or control than the command line client. The only available operations were the high level command's accepted by the command line, i.e. job creation and submission. To get feedback from the operation the calling application had to parse the commands output. The API was implemented only in C++ and thus was unavailable to the large amounts of CMS software written in other languages. As a consequence both CRAB and the Monte Carlo production system ignored the API and used the command line client.

Installation was complicated by the requirement for a MySQL database. This provided a central logging repository but a database server was seen as unnecessary for the average user. It was not possible to automate the installation so this step had to be performed by the user. In principle it was possible for everyone in CMS to share the same database server however firewalls, concerns over a single point of failure and load issues ruled this out in practice. Thus generally either a user or site setup their own database which was used by that user or all users at the site.

The reliability and robustness of the logging and monitoring information were reduced by reliance on a direct connection from the running job back to the MySQL database. This could be blocked by either site with a firewall. If the connection could not be established then neither monitoring nor logging information would be saved. Once the job had finished and the user had access to the output they could run a BOSS command over the journal file and populate the database. This operation required manual intervention, which could not be guaranteed and only provided monitoring information at the end of the job.

## 5.3 New Architecture

Both the CMS analysis and production workflows involved a single computational task split into multiple jobs, each of which was largely identical. Previous BOSS versions were unaware of any relationship or commonalities between jobs, which required the calling application to invoke BOSS separately for each job. Hence it was decided by the author to introduce support for groups of jobs with the task concept. This would allow the calling application, or user, to operate at a higher level of abstraction.

Previously jobs were limited to running a single executable. Clients who required more functionality were forced to submit a script that executed their workflow. This forced the entire workflow to be recorded as a single execution with no monitoring of individual components. It was therefore decided to modify BOSS to run an arbitrary number of fully configurable applications in a single job.

The new default behaviour was to run the job's applications in a linear chain, although more complicated workflows were also supported. It was envisaged that this workflow could be dynamically steered by the conditions on the worker node and the success or failure of the applications. This had many advantages for jobs where pre-programmed actions could be taken if certain error conditions were met. This kind of self-aware job would be increasingly useful as the number of jobs submitted by CMS increased to maintain a manageable number of failures.

The chaining process was complex, with differing user requirements. On account of this complexity it was decided to provide a plug-in mechanism for the chainer. A basic chainer was provided that ran the user's applications in a linear chain. If users required greater functionality they could provide their own. The chainer configuration was inserted into the BOSS task description and passed transparently down to the chainer at run time.

Previously job resubmissions overwrote all information concerning previous executions, resulting in an incomplete activity record. To correct this the logical job concept was separated from physical executions. Within a task jobs could be executed multiple times with the database reflecting this. In the logical view the "job" was renamed "chain" to reflect the idea that more than one application ("program") could run in the workflow. The execution view consisted of "jobs" each composed of multiple "program executions". For each submission of a "chain" the database contained a different "job". The user could query the "chain" and be provided with a complete history of the submitted "jobs".

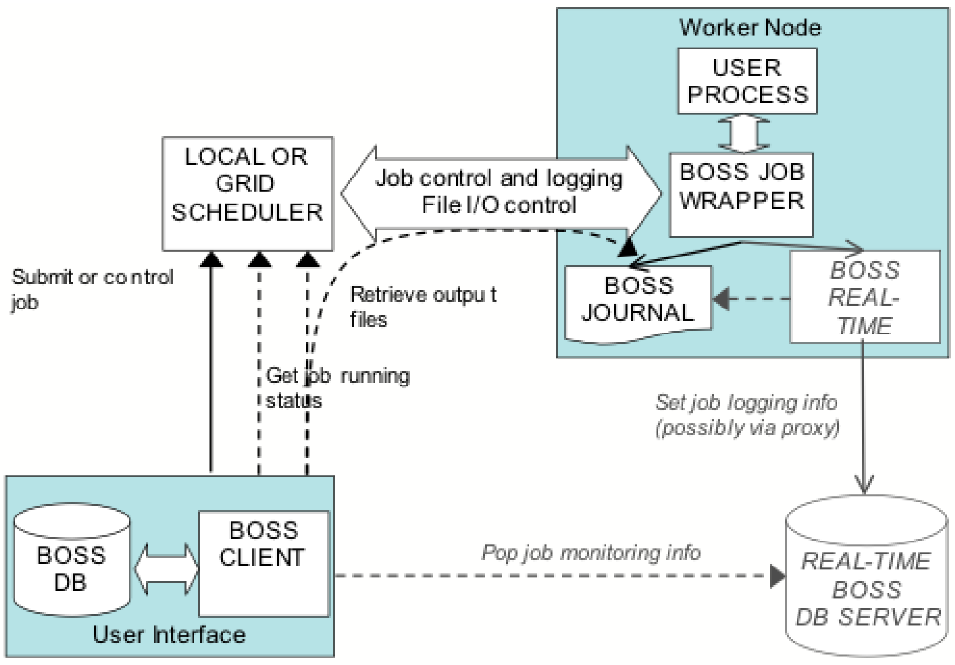


Figure 5.1: New BOSS information flow. [2]

Previously there had been no difference between logging and real time monitoring information. Both were stored in the database via a direct connection from the running job. This was unreliable, unscalable and ignored the differences between logging and real time monitoring. Hence it was decided to separate these different types of information. Logging information availability and integrity were guaranteed while monitoring was optional and collected if possible.

As the logging database was no longer required to accept connections from running jobs the need for a database server was eliminated. The central MySQL server was replaced by a plug-in mechanism that allowed any SQL-compatible database to be used. This provided the opportunity to use embedded databases. Embedded databases used the local filesystem for data storage but had no server component: access was only available via an application or API on the local machine. The new default was to use an embedded database this simplified the installation process and eliminated a potential security vulnerability. Due to the plug-in mechanism users could still, if they wished, use a database server.

Logging information could only be added to the logging database by the BOSS client tools. Logging information from a job was obtained from the journal file and was retrieved at the end of the job. The same BOSS command that retrieved output from a job parsed the journal file and updated the logging database. This was automatic and required no manual intervention by the user.

Monitoring was optional and the information returned via a monitoring database server. These databases could be installed by the individual user or, more likely, by a Tier-1/2 for all its users. Information was sent from jobs to the dedicated monitoring server via a specialist monitoring framework. Multiple frameworks were available in-

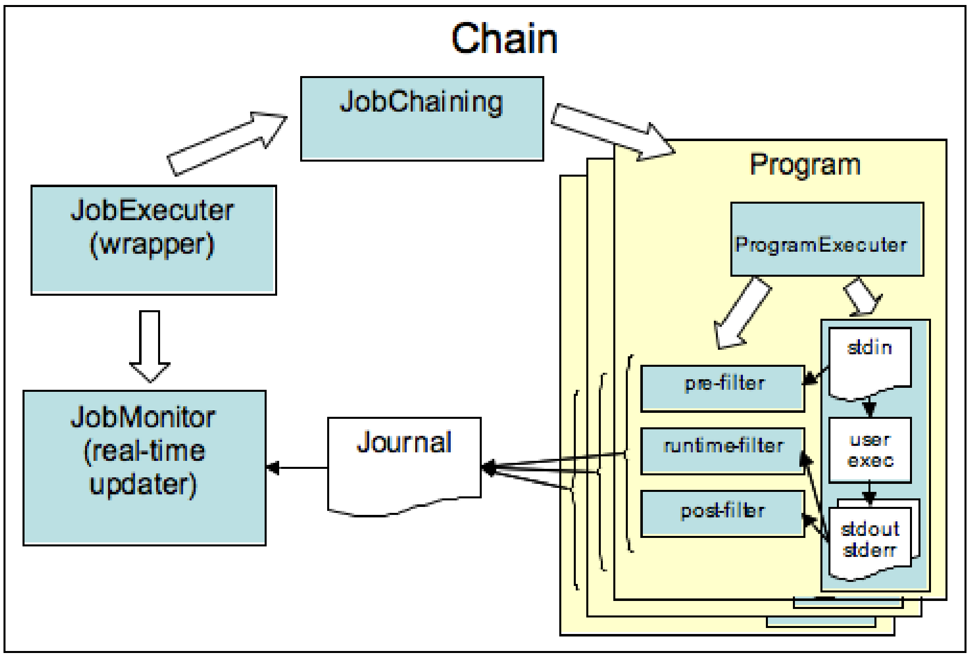


Figure 5.2: BOSS runtime components. [2]

cluding a custom protocol [2] and MonaLisa [65]. Information would then reside in this database until retrieved by the user with the BOSS client. The monitoring information had a limited lifetime and could not be used to populate the logging database.

Figure 5.1 shows the new BOSS components and information flow. The constituent jobs in the task were submitted via a batch scheduler to a worker node. Once the job started BOSS activated the monitoring and chainer. The chainer took responsibility for the users workflow, as illustrated in Figure 5.2. Logging and monitoring information were written to the journal file and, if requested by the user, sent to the monitoring database. The user BOSS client tools retrieved information from the monitoring database and hence allowed users to track their jobs. Once the job finished the user retrieved the output and BOSS updated the logging database from the journal file.

## 5.4 Task Definitions

To create a task the user required a syntax that could express the hierarchy. A logical choice for this was XML, which naturally provides for nested hierarchies of tasks, jobs and applications. Figure 5.3 shows an example task specification. This shows a task composed of 100 chains each composed of one program. The "iterator" concept was introduced to simplify writing specifications. This provided a looping mechanism within the specification. The iterator was replaced by the required number of chains (or other inner element) and the iterator name was replaced by its value as of that loop iteration.

There were two standard XML parsing mechanisms: Document Object Model (DOM) and Simple Api for Xml (SAX). DOM provided a navigable object tree representing the document. SAX provided an event-driven system for calling a given function when a cer-

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<task schema="schema.xml">
  <iterator name="ITR" start="0" end="100" step="1">
   <chain scheduler="glite" rtupdater="mysql" ch_tool_name="jobExecutor">
     <program exec="test.pl"
              args="ITR"
              stderr="err_ITR"
              program_types="test"
              stdin="in"
              stdout="out_ITR"
              infiles="Examples/test.pl,Examples/in"
              outfiles="out_ITR,err_ITR"
              outtopdir="" />
   </chain>
  </iterator>
</task>
```

Figure 5.3: An example XML task specification showing a task composed of 100 chains of one program. [2]

tain tag was encountered. For instance when a task definition was encountered, a function would be called to instantiate a task object. There were advantages and disadvantages to both methods: SAX was generally faster but required development of functions to create tasks and jobs (which would need updating with any schema changes) whereas DOM provided the entire document as an accessible object but had greater memory requirements.

It was decided that the greater speed of SAX was unnecessary for this project as users would generally only create tasks with O(1000) jobs and communication with external services, PubDB and LCG, would dominate any latencies. Due to the predicted size of users tasks the possible higher memory use of DOM was not considered a problem. Thus it was decided to use DOM with the document object as the internal task data structure. This was an idea taken from web browsers where internally an HTML page is stored as a DOM object. In the future when more users create larger tasks these performance issues may become important however it was viewed that in the short-medium term these performance issues were not be important.

The iterator concept required either support from the task object or modification of the XML before parsing because XML does not support the concept of iterators. The latter could be achieved by use of Extensible Stylesheet Language Transformations (XSLT). This is a language capable of describing transformations of an XML document and is usually used to convert XML to viewable HTML. XSL transforms were designed to execute certain actions when a particular XML structure was encountered, but XSL lacks support for loop structures and did not fully support variable modification. The combination of these resulted in a limit of one iterator tag. If more than one tag was included in a document the two would interfere and only the values of one would be correctly replaced. This was not regarded as a major limitation because the most common usage was a single iterator of chains within a task.

## 5.5 API

The previous BOSS API provided the same commands as the command line client with no fine-grained control or feedback. It was decided that a more complete API that allowed greater control and functionality was required.

The API needed to provide functionality suitable for CMS analysis, production and individual users. The API was split into two: user commands (e.g. submit, query) and administration tasks (e.g. register scheduler, job types). The administrator commands could only be run by someone with database administrator privileges.

As with the rest of BOSS these API's were written in C++. However, both the CMS production and analysis frameworks were written in Python. It is possible for a Python program to call a C++ API but this required significant effort on the part of the client developer. As a number of CMS applications, both current and proposed, were written in Python it was decided to provide a Python version of the API's. It was felt that the best way to do this was to wrap the C++ API hence ensuring that the two were continually synchronised.

The wrapping code had to provide a number of features, including:

- Input argument validation;
- Conversion of input Python arguments to C++ datatypes;
- Invocation of the corresponding C++ method; and
- Conversion of output values to Python datatypes.

Advanced C++ features such as polymorphism and exception handling required more complex wrapping code. This problem had been encountered before in computer science and a number of tools had been developed to automate the process. These included SWIG [66], SIP [67] and BOOST.Python [68]. These tools could generate Python code from the C++ sources and their use was less error prone and faster than writing the code manually.

Each tool implemented a slightly different wrapping strategy. SWIG generated wrapper code from the C++ header files that was then built against the C++ libraries. It had support for multiple scripting languages including Python, Perl, Ruby and many others, and was widely adopted with good documentation.

SIP was originally designed to generate Python bindings for a graphical user interface. It used a similar method to SWIG with separate wrapper code generation and build steps. It had the disadvantages that it was specifically designed for one application and not as a general-purpose tool. It also only worked with Python and had minimal documentation.

BOOST.Python took a different approach, called template metaprogramming, where each function was declared in Python with a special syntax. By calling this function

BOOST automatically called the appropriate C++ code. This required a modern compiler to work correctly and only supported the Python language.

Due to the documentation and the support for multiple languages it was decided to use SWIG to create the API bindings. The wrapping added a significant overhead to any function call and SWIG was considerably slower than the other tools [69]. However this overhead was expected to be small compared to the overheads associated with database or scheduler communication as a typical session involved O(10) Python method calls.

Only the generation of the Python bindings required SWIG, while the building and linking did not. The generation step was performed by a developer during the building of a release. With the generated code added to the release the user only needed to build everything together. Thus there was no requirement for SWIG to be available on the user's machine.

## 5.6 Conclusion

Use by both the CMS production and analysis systems required a number of enhancements to BOSS. These included the task concept and better logging mechanisms. An improved API was introduced with greater functionality and multi-language support. To support the task structure a new XML-based task specification was developed.

# Chapter 6

# Higgs mechanism and Supersymmetry

## 6.1 Introduction

The Standard Model (SM) of particle physics has proved to be one of the most successful theories in physics. There are many good overviews of the SM, including [70] and [71]. This chapter focuses on one of the less satisfactory elements of the Standard Model the origin of particle mass, for reasons that will be discussed later. It describes a proposed solution, concentrating on elements that are pertinent to the experimental study presented at the end of this thesis.

The Standard Model describes spin-1/2 matter particles, known as fermions, and their interactions mediated via the exchange of spin-1 particles, known as bosons (with the exception of gravity, which is mediated via a spin-2 boson). The bosons include the photon ($\gamma$), which mediates the electromagnetic force, the $\mathrm{W}^{\pm}$ and Z, responsible for the weak force, gluons, responsible for the strong force, and gravitons that mediate gravity.

In the Standard Model particle wavefunctions are invariant under local phase (gauge) transformations, a principle known as local gauge symmetry. The requirement for fermions to be invariant under the U(1) local gauge transformation results in the coupling of fermions and massless photons, i.e. electromagnetism. Similarly SU(2) and SU(3) invariance forms the basis of the weak and strong forces.

The Standard Model unifies the electromagnetic and weak forces with the electroweak theory of Glashow, Weinberg and Salam [72]. This theory states that both forces are representations of the same fundamental process. Two new charges are introduced, weak isospin (T) and hypercharge (Y), which together with four new fields, $\mathrm{W}_{\mu}^{1,2,3}$ and B are invariant under $\mathrm{SU(2)_L} \times \mathrm{U(1)_Y}$, where L signifies that only left-handed particles take part in the weak interaction. These fields mix to form the $\gamma$ (represented by the field A), $\mathrm{W}^{\pm}$ and Z:

$$A_{\mu} = B_{\mu} \cos\theta_W + W_{\mu}^3 \sin\theta_W \tag{6.1}$$

$$W^{\pm}_{\mu} = \frac{1}{\sqrt{2}}(W^1_{\mu} \mp W^2_{\mu}) \tag{6.2}$$

$$Z_{\mu} = -B_{\mu} \sin\theta_W + W^3_{\mu} \cos\theta_W \tag{6.3}$$

where $\theta_W$ is the Weinberg angle.

The local gauge invariance that introduced the massless photon has a similar effect on all the bosons and thus requires them all to be massless. However experiment has shown that both the $\mathrm{W}^{\pm}$ and Z are massive. The introduction of explicit mass terms for these fields destroys gauge invariance, which is the foundation of the SM. The Higgs mechanism [3] represents a possible solution to this problem.

## 6.2 Higgs mechanism

Boson masses may be introduced through a process known as spontaneous symmetry breaking, which occurs when the Lagrangian is invariant under local gauge transformations but the ground (vacuum) state is not. The electroweak $\mathrm{SU}(2) \times \mathrm{U}(1)_{\mathrm{Y}}$ symmetry must be broken to give mass to the $\mathrm{W}^{\pm}$ and Z whilst leaving the $\gamma$ massless. This can be accomplished with the addition of a complex scalar doublet, $\phi$:

$$\phi = \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix} \tag{6.4}$$

The standard model Lagrangian gains additional terms:

$$\mathcal{L} = D_{\mu}\phi^* D_{\mu}\phi - V(\phi) = D_{\mu}\phi^* D_{\mu}\phi + \mu^2\phi^2 - \lambda\phi^4 \tag{6.5}$$

with the covariant derivative

$$D_{\mu}\phi = (\partial_{\mu} + igW^a_{\mu}T^a + ig'B_{\mu})\phi \tag{6.6}$$

where $T^a$ is the SU(2) group generator and g(g′) is the $W^a_{\mu}$ ($B_{\mu}$) coupling constant. $\mu$ and $\lambda$ are the $\phi$ mass parameter and self-interaction, respectively. The first term of Equation 6.5 describes the kinetic part of the Lagrangian and the last two terms the potential, V($\phi$).

Two possible potential forms exist depending on the value of $\mu^2$, as shown in Figure 6.1. In the case of $\mu^2 > 0$ the potential has a minima at $\phi = 0$. If $\mu^2 < 0$ the minima form a circle where

$$\phi^2 = \frac{\mu^2}{2\lambda} = \frac{v^2}{2} \qquad \text{where}\, v = \frac{\mu}{\sqrt{\lambda}} \tag{6.7}$$

We can then choose a vacuum expectation value (vev) for this field. The vev should be zero for the charged component of the $\phi$ field to preserve the electromagnetic, $\mathrm{U}(1)_{\mathrm{EM}}$, symmetry. Expanding $\phi$ arbitrarily about its vev results in one massive and three massless

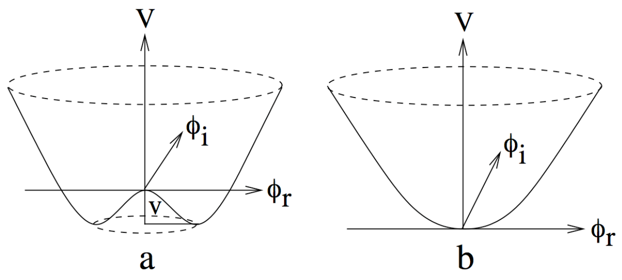


Figure 6.1: The Higgs potential as a function of a complex scalar field with negative (a) and positive (b) mass parameter. $\phi_r$ ($\phi_i$) represents the real (imaginary) component of $\phi$.

bosons. The massless bosons are a result of the Goldstone theorem, which states that for each broken symmetry, a massless field will result. These particles are known as Goldstone bosons and the massive boson is known as the Higgs boson.

As the expansion is arbitrary we can choose the gauge such that the Goldstone bosons are eliminated. We can expand $\phi$ as:

$$\phi(x) = e^{iT^a\theta_a(x)}\frac{1}{\sqrt{2}}\begin{pmatrix}0\\ v+H(x)\end{pmatrix} \tag{6.8}$$

Taking Equations 6.8 and 6.5 together with the physical electroweak fields 6.1, 6.2 and 6.3, we obtain the masses:

$$m_W = m_Z\cos\theta_W = \frac{gv}{2}, \qquad m_A = 0 \tag{6.9}$$

Thus the Higgs mechanism, by utilising spontaneous symmetry breaking and a convenient choice of gauge, gives mass to the W$^\pm$ and Z while keeping the $\gamma$ massless.

The Standard Model also requires a mechanism for explaining the fermion masses. These can be introduced by forming interaction terms between left-handed fermions, right-handed fermions and the Higgs field. These interactions are known as Yukawa couplings. However, for each interaction a new coupling constant is required, the values of which are not predicted.

More of a problem with the Higgs mechanism is that of the Higgs mass itself. The physical mass ($m_H$) is related to the the bare mass ($m_{H_0}$) with corrections from loop diagrams ($\Delta m_H$), as illustrated in Figure 6.2.

$$m_H^2 = m_{H_0}^2 + \Delta m_H^2 \tag{6.10}$$

The fermion contribution to the correction is:

$$\Delta m_H^2 \approx O(-\Lambda^2) \tag{6.11}$$

where $\Lambda$ represents the scale where new physics is introduced and the theory no longer holds. If no new physics occurs at high energy scales then the cut-off becomes the Planck

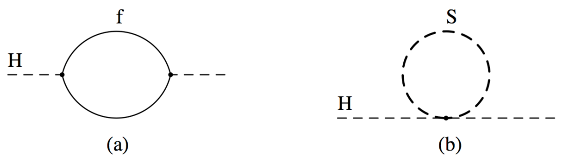


Figure 6.2: Corrections to the Higgs mass from fermion (a) and boson (b) loops. [4]

scale, $10^{19}$ GeV/c$^2$. An upper limit on the Higgs mass can be determined from the WW scattering crosssection which at high energies becomes greater than unitarity for $m_H >$ 1 TeV/c$^2$ [73]. To cancel the correction and result in the Higgs boson mass of $\sim$ 1 TeV the bare mass must be $\approx 10^{19}$ GeV/c$^2$. While mathematically possible this is aesthetically displeasing and is known as the fine tuning problem.

As well as corrections from fermions, boson loops also contribute. Their contribution takes the same form as the fermion loops but with the opposite sign. Thus if a theory could be formed that matches fermions and bosons the correction would be:

$$\Delta m_H^2 \approx O(m_B^2 - m_f^2) \tag{6.12}$$

which is less than $m_H^2$ if the fermion and boson partners have similar masses:

$$m_B^2 - m_F^2 < 1\,\text{TeV/c}^2 \tag{6.13}$$

This indicates new physics below 1 TeV that may take the form of the theory described above. One such theory is supersymmetry.

## 6.3 Supersymmetry

Supersymmetry (SUSY) is a theory that relates fermions to bosons. An operator (Q) is proposed that transforms the two types:

$$Q|boson\rangle = |fermion\rangle \qquad Q|fermion\rangle = |boson'\rangle \tag{6.14}$$

As derived in [4] the Q operator obeys the following anti-commutation rule

$$\{Q_\alpha, Q_\beta^\dagger\} = 2\sigma^\mu_{\alpha\beta} P^\mu \tag{6.15}$$

where $P^\mu$ is the generator of the space-time translation and $\sigma$ the Pauli matrices. Supersymmetric states are grouped together into supermultiplets, which contain a fermion and boson. These are related by the Q operator, with the particles known as each other's superpartner. It can be shown that the mass operator $P^2$ commutes with Q and thus that the superpartners have equal mass [4]. Also the Q operator commutes with the gauge symmetries therefore the superpartners must have equal charge, weak isospin, hypercharge and colour.

It is clear that no superpartners of any known particles have been discovered with the same mass and quantum numbers except spin, hence supersymmetry must be broken. This is possible because the superpartners of the Standard Model particles may have an explicit mass term in their Lagrangian without violating electroweak symmetry [4]. The Supersymmetry breaking mechanism is not understood and most phenomenological models parameterize it.

## 6.4 The Minimal Supersymetric Standard Model

The Minimal Supersymmetric Standard Model (MSSM) is the simplest supersymmetric extension to the standard model that aims to minimise the numbers of superfields and interactions. For each SM particle a new superpartner is required.

Two Higgs doublets with different hypercharge are required to give mass to the up and down type fermions:

$$\phi_u = (\phi_u^+, \phi_u^0) \qquad \phi_d = (\phi_d^0, \phi_d^-) \tag{6.16}$$

The vev of these fields may be written:

$$\langle \phi_u \rangle = \begin{pmatrix} 0 \\ v_u \end{pmatrix} \qquad \langle \phi_d \rangle = \begin{pmatrix} v_d \\ 0 \end{pmatrix} \tag{6.17}$$

These can be related to the $\mathrm{W}^{\pm}$ mass via:

$$v_u^2 + v_d^2 = \frac{2m_W^2}{g} \approx (174\,\mathrm{GeV/c^2})^2 \tag{6.18}$$

The ratio of the two vev's is not predicted, and we define this ratio as:

$$\tan\beta = \frac{v_u}{v_d} \tag{6.19}$$

After $\mathrm{SU(2)} \times \mathrm{U(1)_Y}$ symmetry breaking three of the Higgs degrees of freedom are taken by the $\mathrm{W}^{\pm}$ and Z, leaving five massive Higgs bosons: a charge parity (CP) symmetry odd neutral scalar $A$, two CP even neutral scalars $h$ and $H$ and a pair of charged scalars $H^+, H^-$:

$$A = \sqrt{2}(\cos\beta\ \mathrm{Im}[\phi_u^0] + \sin\beta\ \mathrm{Im}[\phi_d^0]) \tag{6.20}$$

$$H^{\pm} = (\cos\beta\ \phi_u^{\pm} + \sin\beta\ \phi_d^{\mp *}) \tag{6.21}$$

$$\begin{pmatrix} h \\ H \end{pmatrix} = \sqrt{2} \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \mathrm{Re}[H_u^0] - v_u \\ \mathrm{Re}[H_d^0] - v_d \end{pmatrix} \tag{6.22}$$

The mixing angles $\alpha$ and $\beta$ are determined at tree level by:

$$\cos^2(\beta - \alpha) = \frac{m_h^2(m_Z^2 - m_h^2)}{m_A^2(m_H^2 - m_h^2)} \tag{6.23}$$

Theoretical constraints can be applied to $\beta$ as the fermions masses are determined by Yukawa couplings. The requirement that fermion masses do not diverge leads to the constraint $1.2 \lesssim \tan\beta \lesssim 60$. [74]

At tree level the Higgs masses depend only on two parameters, $m_A$ and $\tan\beta$:

$$m_{H^\pm}^2 = m_A^2 + m_W^2 \tag{6.24}$$

$$m_{h,H}^2 = \frac{1}{2}\left(m_A^2 + m_Z^2 \mp \sqrt{(m_A + m_Z)^2 - 4m_Z^2 m_A^2 \cos 2\beta)}\right) \tag{6.25}$$

From 6.25 one can obtain an upper bound on $m_h$:

$$m_h \leq m_Z |\cos 2\beta| \tag{6.26}$$

However loop corrections modify this. The maximum one loop contribution is

$$m_h^2 \lesssim m_Z^2 + \frac{3g^2 m_t^4}{8\pi^2 m_W^2}\left[\ln\left(\frac{M_S^2}{m_t^2}\right) + \frac{X_t^2}{M_S^2}\left(1 - \frac{X_t^2}{12M_S^2}\right)\right] \tag{6.27}$$

where $M_{\text{SUSY}} = M_{\tilde{t}} = M_{\tilde{b}}$, $M_S^2 = M_{\text{SUSY}}^2 + m_t^2$ and $X_t = (A_t - \mu \cot\beta)$.[1] $\mu$ is the Higgs mixing parameter and $A_t$ the trilinear Higgs-$\tilde{s}$ coupling. This correction increases with $X_t$ reaching its maximal value, $m_h^{max}$, at $X_t = \sqrt{6}M_S$:

$$m_h^{max} \lesssim 130\,\text{GeV/c}^2 \tag{6.28}$$

This number is calculated with a top mass of $175\,\text{GeV/c}^2$ and assuming that no super-partner masses ($M_{\text{SUSY}}$) exceed $1\,\text{TeV/c}^2$.

Two Higgs mass regimes may be identified depending on the ratio of $m_A$ and $m_h^{max}$:

- $m_A \gg m_h^{max}$: $m_A \sim m_H \sim m_{H^\pm}$ with h behaving like the SM Higgs.
- $m_A \ll m_h^{max}$: $m_A \sim m_h$ with H behaving like the SM Higgs.

The couplings of the MSSM Higgs sector compared to the SM are shown in Table 6.1. It can be seen that the h and H bosonic couplings are greatly suppressed compared to the SM couplings. In the two mass regimes the couplings simplify. For large $m_A$ one can derive from Equation 6.23 that $\cos(\beta - \alpha) \ll 1$. This implies that $h$ coupling to bosons will be similar to the SM Higgs boson and the boson couplings of the $H$ will be strongly suppressed, while the coupling of both the $A$ and $H$ will be enhanced by a factor $\tan\beta$ for down-type fermions. This may prove useful in experimental searches where down-type

[1] A tilde indicates the superpartner of the indicated particle.

| coupling | $h$ | $H$ | $A$ |
|---|---|---|---|
| up-type fermions | $\cos\alpha/\sin\alpha$ | $\sin\alpha/\sin\beta$ | $\cot\beta$ |
| down-type fermion | $-\sin\alpha/\cos\beta$ | $\cos\alpha/\cos\beta$ | $\tan\beta$ |
| W,Z | $\sin(\beta-\alpha)$ | $\cos(\beta-\alpha)$ | 0 |

Table 6.1: MSSM couplings for the neutral Higgs bosons.

final state particles are sought. This is especially true in the case of b quarks and $\tau$ leptons since the Higgs coupling is proportional to mass.

## 6.5 Summary

The Higgs mechanism provides a means for giving mass to the $W^{\pm}$ and Z bosons. Various problems have been identified with the Higgs mechanism that supersymmetric theories aim to solve. The MSSM is the simplest supersymetric extension to the Standard Model and the phenomenology of its Higgs sector has been explained. Neither Higgs bosons nor any supersymmetric particles have so far been discovered. They are the prime targets of the next generation of particle experiments and if they exist with masses $< 1\,\text{TeV/c}^2$ they should be seen at LHC.

# Chapter 7

# $\tau$ tagging and calibration

## 7.1 Introduction

The $\tau$ lepton will be a useful signature of many physics processes at CMS. The $\tau$ decays via the weak mechanism with a characteristic decay time of $\tau = 290 \times 10^{-15}$s ($c\tau = 87.11$ $\mu$m) [75]. The $\tau$ primarily decays hadronically with a branching ratio of 65%, forming a $\tau$ jet. When the jet has large transverse momentum, $p_T \gg M_\tau$ (1.78 GeV/c$^2$), the constituent hadrons' momentum transverse to the jet axis is small resulting in a highly collimated, narrow jet. 77% of hadronic decays consist of a single charged hadron and a number of neutral pions (one prong decays). The majority of the remainder of the hadronic decays involve 3 charged particles(three prong decays). Decays with 5 charged particles also occur but are much rarer.

The standard CMS reconstruction cone size (R = 0.5) was too large for $\tau$ jets, introducing detector noise. Consequently this was optimised during this study. Figure 7.1 shows the energy containment and resolution, $\sigma(E)/ < E >$, for $\tau$ jets as a function of reconstruction cone size for various $E_T^{MC}$ ranges. These plots show that a cone size of 0.4 was appropriate for $\tau$ jets, with a 98% energy containment and good resolution.

## 7.2 $\tau$ tagging

A number of $\tau$ tagging methods have been developed for CMS [9, 76]. Some of the most common techniques are presented below.

### 7.2.1 Tracker isolation

The tracker isolation method sought a narrow isolated $\tau$ jet. The principle is shown in Figure 7.2. Tracks with $p_T$ greater than a defined value, $p_T^m$, with respect to the beam were sought within a matching cone of radius $R_m$ about the calorimeter jet axis. The highest $p_T$ track with unsigned transverse impact parameter $IP_T < 300$ $\mu$m was defined to be the leading track ($tr_1$). The $IP_T$ cut removed fake tracks reconstructed from hits left by soft particles in QCD jets 7.3. Tracks within the cone $R_S$ centred on $tr_1$ with a z-impact

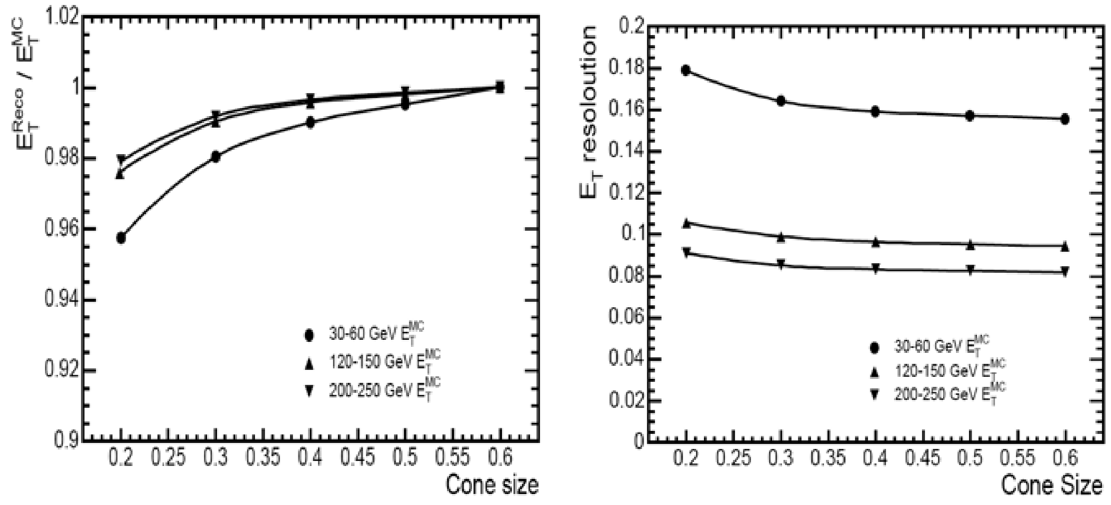


Figure 7.1: The $\tau$ jet energy scale, normalised to that obtained with a 0.6 cone, (left plot) and resolution, $\sigma/ < E >$, (right plot) as a function of reconstruction cone size. [9, 76]

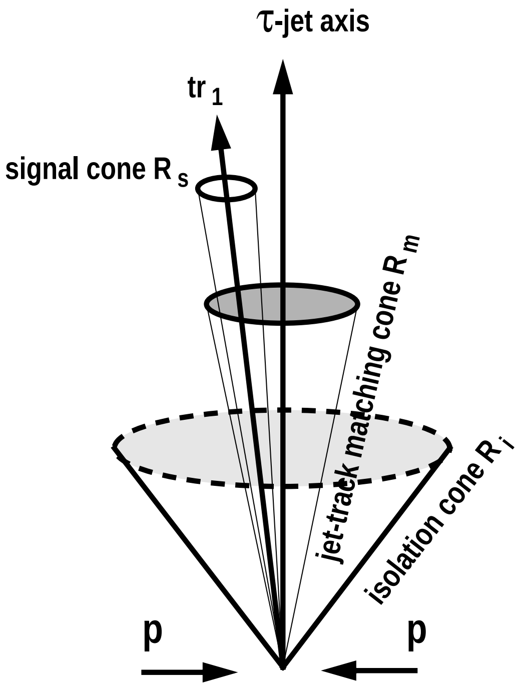


Figure 7.2: $\tau$ identification method using the tracker. [9, 76]

parameter $z_{tr}$ close to the z-impact parameter of the lead track $z_{tr}^{ltr}$ ($|z_{tr} - z_{tr}^{ltr}| < \Delta z_{tr}$) were assumed to come from the $\tau$ and termed "signal tracks". Other tracks within the cone $R_i$ with $P_T$ w.r.t the beam greater than $P_T^i$ and satisfying $|z_{tr} - z_{tr}^{ltr}| < \Delta z_{tr}$ were termed "isolation tracks". The isolation criteria required zero isolation tracks.

Figure 7.4 shows the tracker isolation efficiency for both $\tau$ and QCD jets. Jets were reconstructed using an iterative cone algorithm with a cone size of 0.4. Tracks were reconstructed with the combinatorial track finder algorithm (the standard CMS tracking algorithm [11]) and were required to have at least 8 reconstructed hits, 2 in the pixel

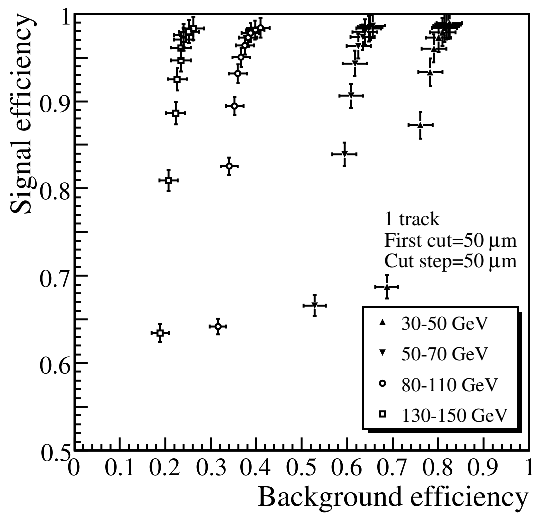


Figure 7.3: $\tau$ tagging efficiencies for 1-prong $\tau$ jets and 4 $E_T^{MC}$ bins of 1-prong QCD multi jets. $E_T^{MC}$ obtained with Monte Carlo particle jets excluding neutrinos. The upper cut on transverse impact parameter of the lead track is varied between 50 – 550 $\mu$m. [9, 76]

| QCD jets; $E_T^{MC}$ (GeV) | 30–50 | 50–70 | 80–110 | 130–150 |
|---|---|---|---|---|
| 1 track | 63 % | 72 % | 69 % | 60 % |
| 3 tracks | 7 % | 9 % | 9 % | 13 % |
| 1 or 3 tracks | 70 % | 81 % | 78 % | 73 % |
| $\tau$ jets; $E_T^{MC}$ (GeV) | 30–50 | 50–70 | 80–110 | 130–150 |
| 1 track | 81 % | 77 % | 71 % | 70 % |
| 3 tracks | 10 % | 16 % | 16 % | 20 % |
| 1 or 3 tracks | 91 % | 93 % | 87 % | 90 % |

Table 7.1: The efficiency of the track counting requirement for $\tau$ and QCD jets in different bins of $E_T^{MC}$. [9, 76]

detector, and a $\chi^2 < 10$. Tracker isolation parameters used were: $R_m = 0.1$, $p_T^i = 1$ GeV/c, $\Delta z_{tr} = 2$ mm and the leading track $p_T > 6$ GeV/c. It can be seen that the efficiency of genuine $\tau$ jets was independent of $R_i$, for the studied range, whereas QCD multi-jet rejection increased with isolation cone size.

The well-defined number of charged particles in a $\tau$ decay lead naturally to a requirement on the number of signal tracks of 1 or 3. The effect of this can be seen in Table 7.1. From this it was concluded that the requirement on the number of signal tracks did not significantly improve QCD multi-jet rejection vs signal efficiency.

### 7.2.2 Electromagnetic calorimeter isolation

$\tau$ jets produce a narrow energy deposit in the electromagnetic calorimeter. The electromagnetic isolation $P_{isol}$ parameter was found to provide the best hadronic jet rejection [9, 76].

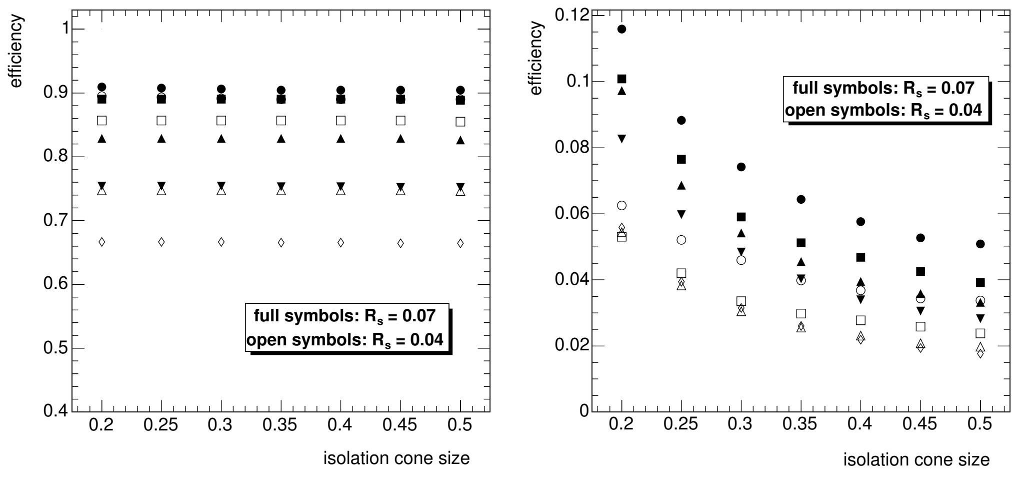


Figure 7.4: The tracker isolation efficiency for $\tau$ jets (left plot) and QCD multi-jets (right plot) as a function of the isolation cone $R_i$ for 2 values of the signal cone $R_S$=0.07 (full symbols) and $R_S$=0.04 (open symbols). In order of decreasing efficiency the symbols correspond to particle level jet $E_T^{MC}$ bins of 130–150, 80–110, 50–70 and 30–50 GeV. The remaining tracker isolation parameters were: $R_m$=0.1, $p_T^i$=1 GeV/c. $\Delta z_{tr}$=2 mm and the leading track $p_T > 6$ GeV/c. [9, 76]

Where $P_{isol}$ was defined as

$$P_{isol} = \sum_{\Delta R<0.40} E_T - \sum_{\Delta R<0.13} E_T \tag{7.1}$$

Jets with $P_{isol} < P_{isol}^{cut}$ were considered $\tau$ candidates. Figure 7.5 shows how the efficiency for $\tau$ and QCD multi-jets jets varied with $P_{isol}^{cut}$. Generally a value of 5 GeV was recommended for offline analysis.

### 7.2.3 Electron rejection

An isolated electron (from a leptonic $\tau$ decay for instance) could pass both the $\tau$ jet tracker and electromagnetic isolation criteria. An offline cut on the most energetic HCAL tower in the jet was introduced to suppress these. Figure 7.6 shows the hottest (maximal $E_T$) HCAL tower for 35 GeV electrons and 2 $p_T$ ranges of $\tau$s. The high electron $E_T$ tail was due to electrons passing through $\eta - \phi$ ECAL gaps and the instrumentation gap between the barrel and endcap. Table 7.2 shows the efficiency of this cut for $\tau$ jets and electrons.

$\mu$ lepton rejection was not investigated as energy deposited in the calorimeter was only of the order of a few GeV, well below the jet $E_T$ cuts used in most CMS analyses.

### 7.2.4 High Level Trigger

The High Level Trigger utilised both ECAL and tracker isolation. The HLT $\tau$ trigger was optimised for the MSSM heavy Higgs boson decaying to 2 hadronic $\tau$ jets.

The recommended HLT algorithm, “Calo+Pxl”, [9, 76] relied on ECAL isolation fol-

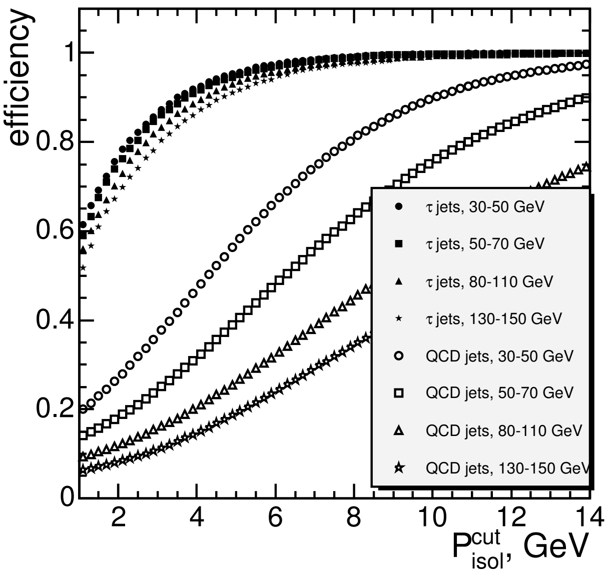


Figure 7.5: Electromagnetic calorimeter isolation efficiencies for $\tau$ and QCD multi-jets as a function of $P^{cut}_{isol}$. [9, 76]

| cut | electron | $\tau$ jet $E_T$ 40–60 GeV | | $\tau$ jet $E_T$ 100–140 GeV | |
|---|---|---|---|---|---|
| | | $p_T^{ltr} > 10$ GeV | $p_T^{ltr} > 25$ GeV | $p_T^{ltr} > 10$ GeV | $p_T^{ltr} > 25$ GeV |
| >1 GeV | 0.08 | 0.936 | 0.971 | 0.977 | 0.991 |
| >2 GeV | 0.03 | 0.854 | 0.917 | 0.942 | 0.969 |

Table 7.2: Efficiency of the transverse energy of the hottest HCAL tower (maximal $E_T$) cut for electrons $p_T = 35$ GeV/c and for different $\tau$ jet $p_T$ ranges and lead track $p_T$ ($p_T^{ltr}$). [9, 76]

lowed by tracker isolation using only the pixel layers. This algorithm took as seeds the 2 Level-1 $\tau$ primitives. If no second candidate was identified the most energetic Level-1 central jet candidate was considered.

Figure 7.7 shows the efficiency of the ECAL isolation criteria for $\tau$ jets and multi-jets. A reduction in the multi-jet rate of $\sim 3$ was achieved with $P^{cut}_{isol} = 5\,$GeV.

Jets that passed the calorimeter isolation were passed through tracker isolation with the pixel detector. Track candidates were formed from hit pairs in the first two pixel layers. Quality cuts were applied consistent with a track $p_T$ of 1 GeV/c or above. Hits from the third pixel layer were combined with the track candidates to form pixel-tracks. A list of primary vertices was formed from the pixel-track z impact parameters. Vertices associated with fewer than three pixel-tracks were discarded. Each vertex was computed as the mean of the z impact parameters of the constituent tracks.

The tracker isolation was then run with the pixel-tracks. The lead track was required to have $p_T^{ltr} > 3\,$GeV/c and to be within the cone $R_m = 0.1$ around the jet axis. All further tracks were required to come from the same vertex as the lead track. Signal tracks were located within the cone $R_S$=0.07 with no further tracks permitted within the isolation cone, $R_i$.

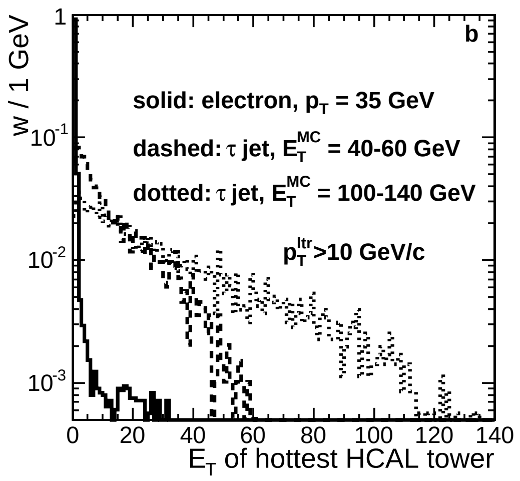


Figure 7.6: Transverse energy of maximal $E_T$ HCAL tower ( GeV). All histograms are normalised to unity. [9, 76]

Figure 7.8 shows the Calo+Pxl single (left) and double (right) $\tau$ efficiencies for both H$\rightarrow \tau\tau$ and QCD multi-jets ($50 < \mathrm{E_T^{MC}} < 170\,\mathrm{GeV}$) as a function of the isolation cone size. The A/H$\rightarrow \tau\tau$ analysis required a double trigger. It can be seen that a double trigger background rejection rate of $\sim 10^{-3}$ could be achieved with an $\mathrm{R_i}$ of 0.45–0.5, giving a signal efficiency of 0.29–0.32.

## 7.3 $\tau$ jet energy scale

Figures 7.9 and 1.9 show the $\tau$ and QCD multi-jet energy scales. The drop at $\eta \simeq 1.4$ was due to the instrumentation gap between the calorimeter barrel and endcap sections. It can be seen that the $\tau$ energy scale was significantly higher than that of multi-jets. This was a consequence of the intrinsically higher electromagnetic response and the $\tau$ jets' low multiplicity. Both these resulted from $\tau$ jets being predominantly light quark jets containing neutral pions. Thus $\tau$ jets required a dedicated energy scale calibration.

### 7.3.1 Monte Carlo calibration

A Monte Carlo $\tau$ jet energy calibration was developed in this study for use in Monte Carlo analyses. This correction was obtained from a parameterisation in $\eta$ and $\mathrm{E_T}$ of the ratio $\mathrm{E_T^{reco}/E_T^{MC}}$ for Monte Carlo single prong, 3 prong and 1 + 3 pronged decays in the presence of pileup at an instantaneous luminosity of $\mathcal{L} = 2 \times 10^{33}\,\mathrm{cm^{-2}\,s^{-1}}$. Figure 7.10 shows the $\tau$ jet energy scale before and after calibration for 1 + 3 pronged $\tau$ jets.

The energy resolution after the correction can be seen in Figure 7.11 and parameterised as:

$$\frac{\sigma\left(\frac{\mathrm{E_T^{corr}}}{\mathrm{E_T^{MC}}}\right)}{\langle\frac{\mathrm{E_T^{corr}}}{\mathrm{E_T^{MC}}}\rangle} = \frac{0.883}{\sqrt{\mathrm{E_T^{MC}}}} \oplus 0.058 \tag{7.2}$$

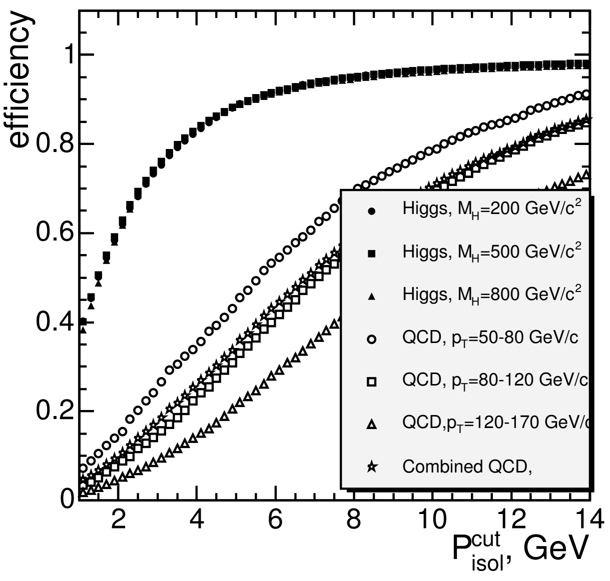


Figure 7.7: The efficiency of the ECAL isolation at the High Level Trigger for the signal and the QCD multi-jet background as a function of the cut $P_{\rm isol}^{\rm cut}$ on the ECAL isolation parameter. [9, 76]

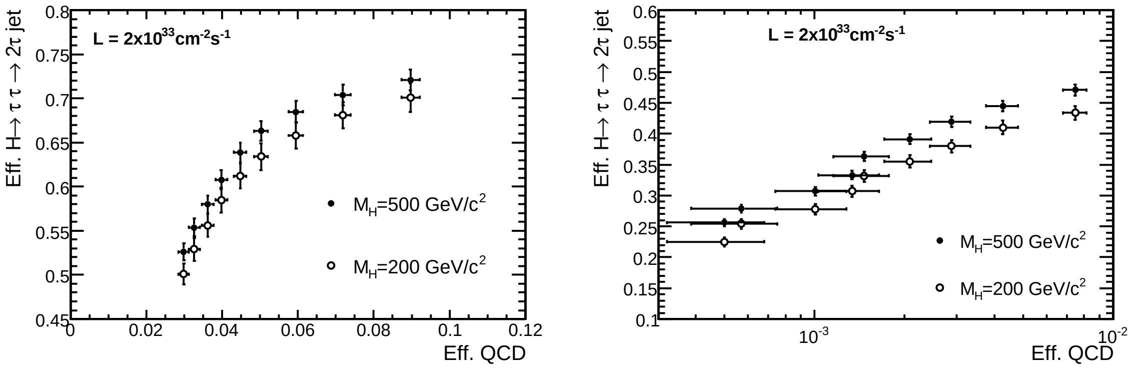


Figure 7.8: Efficiency of Calo+Pxl trigger applied to the first jet (left) and to both jets (right) for signal events versus efficiency for QCD multi-jet events. Two Higgs boson masses of $M_H$=200 and 500 GeV/$c^2$ are shown. The isolation cone is varied from 0.2 to 0.6 in steps of 0.05; the signal cone is 0.07; the matching cone is 0.1; and the $p_T$ of the leading tracks must exceed 3 GeV/$c$. [9, 76]

for $\tau$ jets with $E_T$ between 30 and 300 GeV and pseudorapidity less than 2.3. Figure 1.10 shows the comparative plot for QCD multi-jets.

### 7.3.2 Proposal for calibration from data

Once CMS begins data taking a calibration from real data will be needed. One proposal for this involved taking $Z \to \tau\tau \to l + jet$ events and reconstructing the Z mass peak [9]. The $\tau$ energy scale could be computed from the well known lepton energy scale and Z mass peak. Disadvantages of this approach included background contamination and the

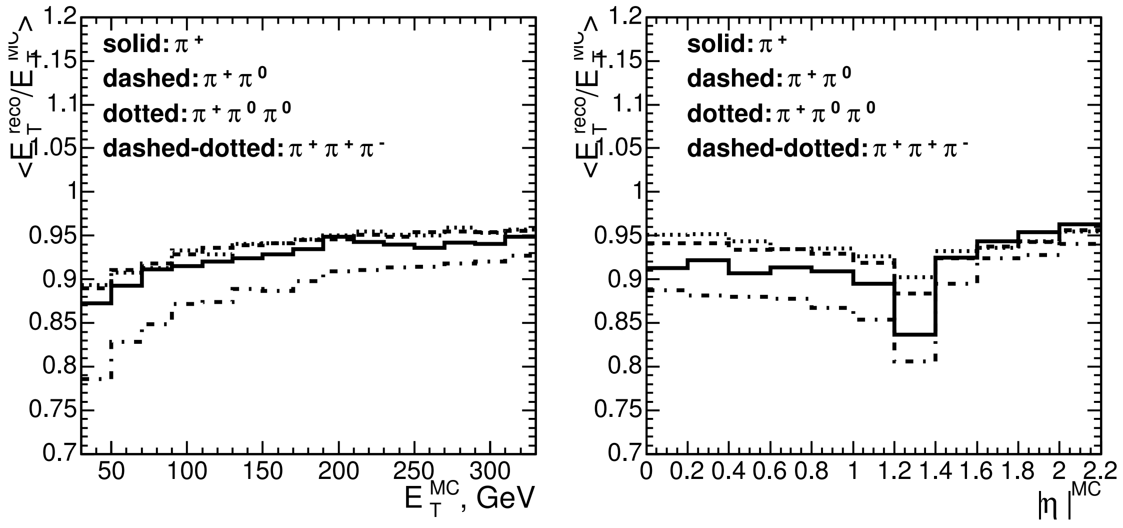


Figure 7.9: The Monte Carlo $\tau$ jet energy scale for different decay final states. [9, 76]

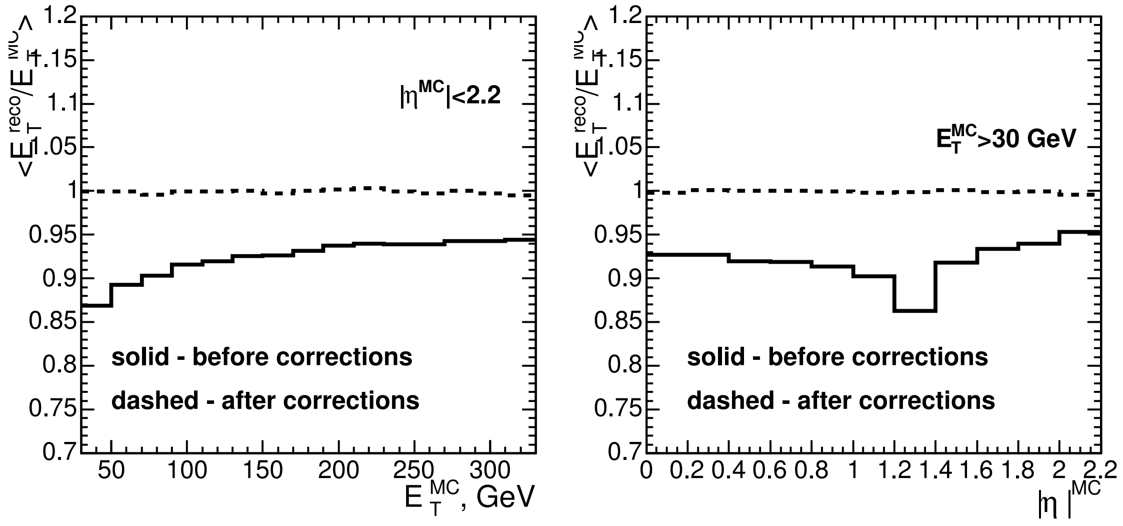


Figure 7.10: Simulated $\tau$ jet $E_T$ scale before and after Monte Carlo corrections for 1 and 3 pronged $\tau$ jets in the presence of low luminosity pileup. [9, 76]

$E_T^{miss}$ resolution.

Another approach was proposed. [9, 76] and studied here, using $\gamma$ + jet events, which are already planned to be used to calibrate the QCD multi-jet energy scale. It has been proposed that those jets that pass the $\tau$ tagging criteria may have an energy scale similar to that of genuine $\tau$ jets. To evaluate the feasibility of this approach the QCD multi-jets sample was passed through the $\tau$ tagging criteria and the energy scale determined. Jets were reconstructed with cone size 0.4 and passed through both tracker and ECAL isolation. Isolation was performed with parameters $P_{isol}^{cut}$=5 GeV, $R_i$=0.4, $R_S$=0.07, lead track $p_T > 10$ GeV/c and with the requirement for 1 or 3 signal tracks. QCD multi-jets that passed all $\tau$ criteria were termed $\tau$ like.

Figure 7.12 shows the results. One can see that $\tau$ like jets had a much higher calorime-

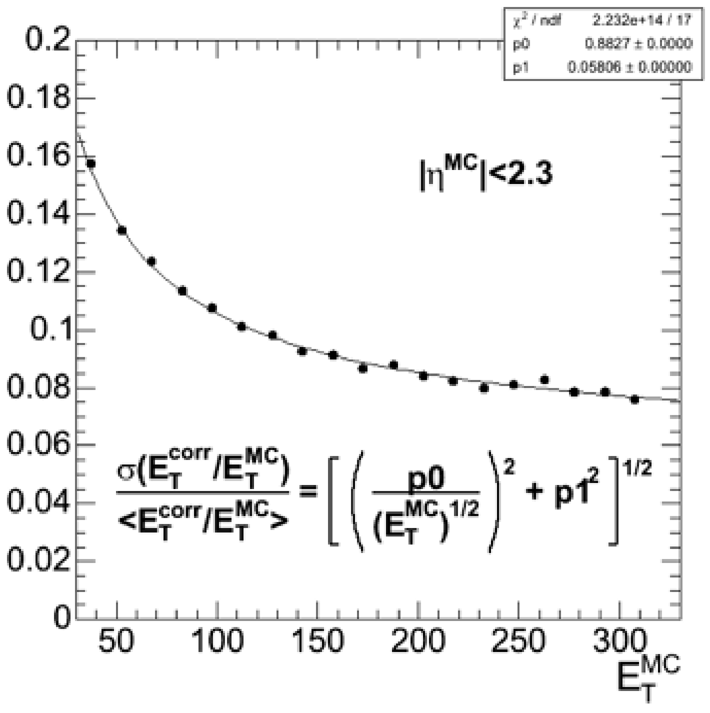


Figure 7.11: The calibrated $\tau$ jet $\mathrm{E_T}$ resolution in the presence of low luminosity pileup. [9, 76]

ter response compared to untagged jets. However there was still a 5–10% systematic offset. This is probably due to the same reasons that give $\tau$ jets there inherently high energy response. It is possible that this difference may be reduced when CMS adopts particle flow algorithms which could be used to select QCD jets that are even more similar to $\tau$ jets than the ones presently selected. If this approach does not provide similar responses an extra correction factor could be applied to the $\tau$-like jet energy scale. However this would introduce new errors and systematics and would therefore be undesirable.

The QCD multi-jet miss-tag efficiency as a function of $\mathrm{E_T}$ is shown in Figure 7.13.

## 7.4 Conclusion

The $\tau$ tagging and calibration performance are important for a number of physics channels at CMS. Multiple $\tau$ tagging methods have been developed for use in the High Level Trigger and offline analysis. A Monte Carlo jet energy calibration has been developed and a method proposed to obtain the calibration from data.

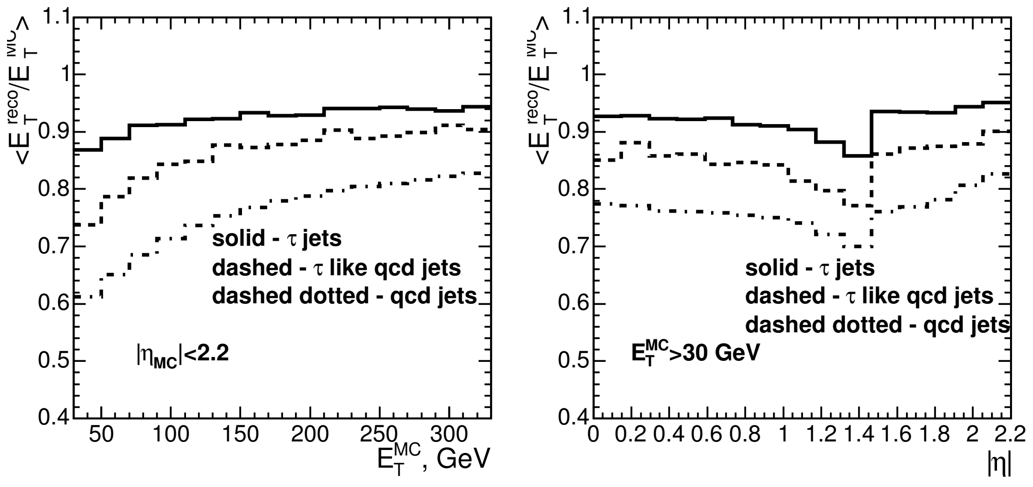


Figure 7.12: The energy scale for $\tau$ jets, QCD multi-jets that pass the $\tau$ tagging criteria and jets which do not. [9, 76]

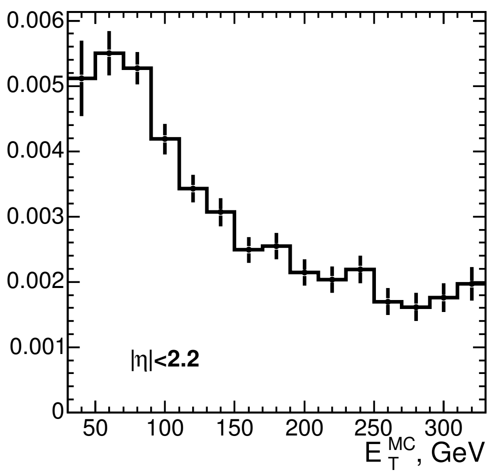


Figure 7.13: The efficiency for a QCD multi-jet to pass the $\tau$ tagging criteria. [9, 76]

# Chapter 8

# Study of A/H→ $\tau\tau$ →two jets

## 8.1 Introduction

Discovery of the heavy neutral Higgs boson would be an important verification of the MSSM Higgs sector and hence the MSSM itself. If calculated its mass could be used as part of a global fit to determine all MSSM parameters.

Negative searches for the Higgsstrahlung process $e^+e^- \to Zh/ZH$ and $e^+e^- \to Ah/AH$ at LEP2 yielded limits of $m_{h,H} > 93\,\mathrm{GeV/c^2}$ [77]. A $\tan\beta$ range could only be excluded within the context of a particular MSSM scenario. The $m_h^{max}$ benchmark scenario, see Section 6.4 where $\mathrm{m_h} \approx 130\,\mathrm{GeV/c^2}$ and $\mathrm{m_A} \gg \mathrm{m_h}$, provided the most conservative $\tan\beta$ exclusion of $0.7 > \tan\beta > 2.0$ for top quark mass $(\mathrm{m_t}) = 174.3\,\mathrm{GeV/c^2}$ [77]. The Tevatron placed upper limits on the $\tan\beta - m_A$ plane, which are shown together with the LEP2 limits in Figure 8.1.

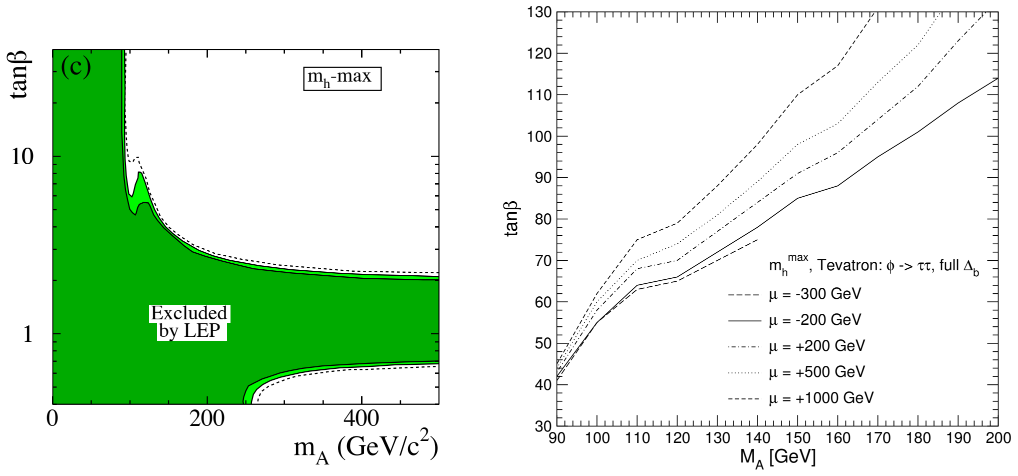


Figure 8.1: LEP2 (left) and Tevatron (right) limits in the $\tan\beta$ - $m_A$ plane for the $m_h^{max}$ scenario. [77, 78]

It is common for an MSSM search to be conducted with a set of benchmark parameters. The scenario that provides the most conservative $\tan\beta$ reach is the $m_h^{max}$ scenario. This scenario was described Chapter 7 and occurs when $m_A \gg m_h^{max}$. In this scenario the light

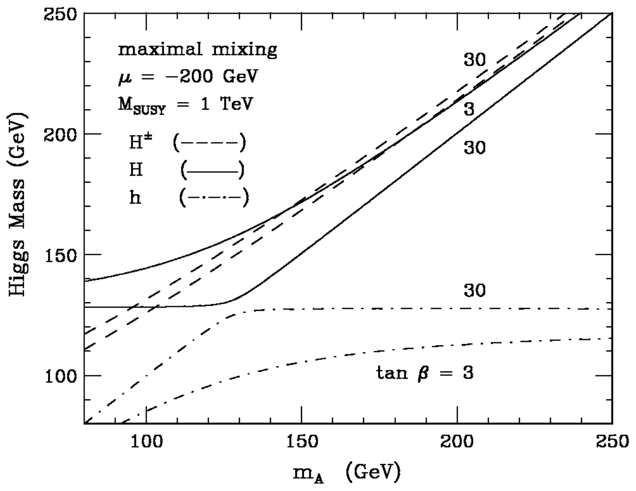


Figure 8.2: Masses of the Higgs bosons for two possible values of $\tan\beta$ as a function of $m_A$ in the $m_h^{max}$ scenario. [79]

scalar, h, reaches its maximal value of $\sim 130\,\mathrm{GeV/c^2}$ and the neutral heavy Higgs bosons, $A$ and $H$, become almost degenerate in mass, this is shown in Figure 8.2. The mass difference between the two is within CMS's jet resolution, thus the two signal distributions are combined in searches.

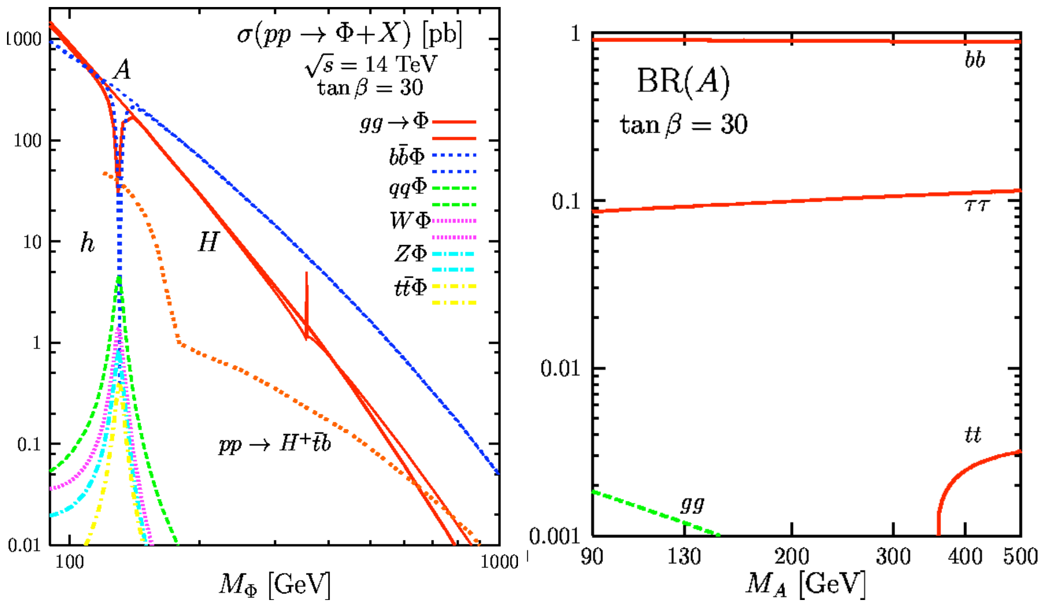


Figure 8.3: Cross section for MSSM Higgs boson production (left) and branching ratio of the $A$ (right) at the LHC for $\tan\beta = 30$. [79]

Figure 8.3 shows the $H/A$ cross-sections and $A$ branching ratio (BR) for the $m_h^{max}$ scenario with $\tan\beta = 30$. Decays to b quarks and $\tau$'s dominate due to the enhanced coupling to heavy down-type quarks, see Table 6.1. Similar branching ratios are also obtained for the $H$. The dominant decay to b quarks would be difficult to search for with CMS due to the large hadronic background at the LHC. Thus the second most common decay, to 2 $\tau$ leptons, was studied.

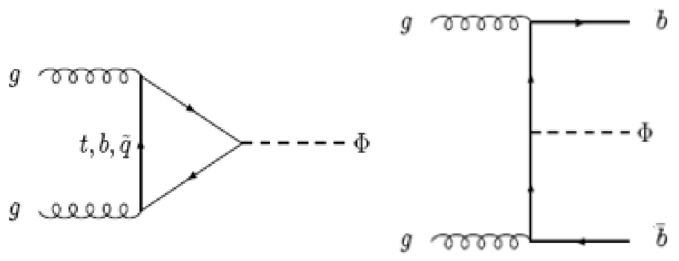


Figure 8.4: Feynman diagrams of the major Higgs production mechanisms studied in this analysis. Gluon-gluon fusion is shown on the left and associated b production on the right.

Many $\tau$ decay channels have been studied at CMS [10] with fully hadronic final states, with $\tau \to \nu + j$ providing the highest $m_A$ reach. This channel relies on the production associated with b quarks [10,80] (Figure 8.4) and uses $\tau$ and b-tagging for event selection. It was decided for this study to investigate the possibility of replacing the b-tagging selection with one based on $\mathrm{E_T^{miss}}$. With the elimination of b-tagging this analysis was also sensitive to the second most common production mechanism, gg → A/H(Figure 8.4). The study was performed for the three sample values of $m_A$ = 200, 500 and 800 GeV/c$^2$. This analysis is described below and the resulting reach in the $\tan\beta - m_A$ plane for the $m_h^{max}$ scenario presented.

## 8.2 Signal and background generation

Until CMS data-taking begins Monte Carlo data has to be used. Events were produced with full detector simulation at an instantaneous luminosity of $\mathcal{L} = 2 \times 10^{33}\,\mathrm{cm^{-2}\,s^{-1}}$. Signal events were generated with PYTHIA [81] processess 181 (gg → bbA/H) and 152 (gg → A/H) for 3 values of $\mathrm{m_A}$: 200, 500 and 800 GeV/c$^2$. Backgrounds included events with $\tau$ jets and those with jets, or electrons, which could be mis-tagged as a $\tau$. PYTHIA-generated backgrounds included QCD multi-jets, $\mathrm{Z}(\gamma) \to \tau\tau$, $\mathrm{Z}(\gamma) \to \mathrm{e^+e^-}$, $\mathrm{t\bar{t}}$, Wt and W + jet. W bosons in the samples $\mathrm{t\bar{t}}$, W + jet and Wt were forced to decay $W \to \tau\nu$. All $\tau$ decays were forced to hadrons with the TAUOLA package [82].

Loose cuts were applied to the background samples at the generator level to avoid generating excess events that would not pass the offline selection. These selections were looser than those used offline. The main requirement was for 2 $\tau$-like jets with generated $\mathrm{p_T}$ ($\mathrm{\hat{p}_T}$) > 50 GeV. A jet was formed with the PYTHIA PYCELL routine and tagged as $\tau$-like if it was within the tracker acceptance, $|\eta| < 2.4$, and had at least one stable charged particle with $\mathrm{p_T}$ > 30 GeV/c. No $\mathrm{p_T}$ preselection was applied to the $\mathrm{Z}(\gamma) \to \tau\tau$ sample. A cut of $\mathrm{M_{ee}} > 15\,\mathrm{GeV/c^2}$ was applied to the $\mathrm{Z}(\gamma) \to \mathrm{e^+e^-}$ sample. No preselection was applied to the signal samples. The generated number of events and generator cut efficiency, $\epsilon_{kine}$, for all generated datasets are shown in Table 8.1.

| sample | $\sigma\times$ BR (fb) | generated luminosity (fb$^{-1}$) | generated events | $\epsilon_{kine}$ |
|---|---|---|---|---|
| gg $\rightarrow$ bbA/H $m_A = 200\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| gg $\rightarrow$ bbA/H $m_A = 500\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| gg $\rightarrow$ bbA/H $m_A = 800\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| gg $\rightarrow$ A/H $m_A = 200\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| gg $\rightarrow$ A/H $m_A = 500\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| gg $\rightarrow$ A/H $m_A = 800\,\mathrm{GeV/c^2}$ | - | - | 100K | 1 |
| QCD $50 < \hat{p}_T < 80$ GeV | $2.11 \times 10^{10}$ | 0.020 | 100K | $2.44\times10^{-4}$ |
| QCD $80 < \hat{p}_T < 120$ GeV | $2.94 \times 10^{9}$ | 0.012 | 200K | $5.77\times10^{-3}$ |
| QCD $120 < \hat{p}_T < 170$ GeV | $5.03 \times 10^{8}$ | 0.009 | 200K | $4.19\times10^{-2}$ |
| QCD 170 GeV $< \hat{p}_T$ | $1.33 \times 10^{8}$ | 0.008 | 1000K | $2.12\times10^{-1}$ |
| Z($\gamma$) $\rightarrow \tau\tau$ $80 < M_Z < 130$ | $1.57 \times 10^{6}$ | 4.3 | 128K | $1.90\times10^{-2}$ |
| Z($\gamma$) $\rightarrow \tau\tau$ $130 < M_Z < 300$ | $1.24 \times 10^{4}$ | 59 | 70K | $9.53\times10^{-2}$ |
| Z($\gamma$) $\rightarrow \tau\tau$ $300 < M_Z$ | $6.22 \times 10^{2}$ | 299 | 60K | $3.23\times10^{-1}$ |
| Z($\gamma$) $\rightarrow e^+e^-$ | $3.96\times10^{6}$ | 0.24 | 985K | 1 |
| $t\bar{t}$ | $5.76\times10^{3}$ | 285 | 80K | $4.88\times10^{-2}$ |
| W + jet | $5.74\times10^{5}$ | 32 | 400K | $2.16\times10^{-2}$ |
| Wt | $7.10\times10^{2}$ | 3053 | 30K | $1.38\times10^{-2}$ |

Table 8.1: Generated datasets listing the process cross section, generation cut efficiency, generated events and luminosity. Luminosity and cross sections are not shown for the signal samples as they cannot be calculated without knowledge of the MSSM parameters. Background cross sections are assumed to have no error. Errors on the number of background events are introduced later for the dominant backgrounds.

## 8.3 Level-1 and high level triggers

The Level-1 $\tau$ trigger was described in Section 1.2.6. Either a single 93 GeV or two 66 GeV $\tau$ triggers were required. These values were optimised for the available trigger bandwidth, 3.2 kHz, as shown in Ref [9]. The $E_T$ cut was applied after Level-1 jet corrections. These corrections were optimised for non $\tau$ jets and thus over-corrected the $\tau$ $E_T$. The signal efficiency and purity as a function of $\tau$ $E_T$ can be seen in Figure 8.5.[1]

The HLT strategy was described in section 7.2.4. The event selection criteria required two HLT $\tau$ candidates with no jet $E_T$ threshold. No $E_T$ threshold was applied as the Level-1 threshold had already been applied and the $\tau$ id criteria resulted in an adequate rate of 4 Hz.

## 8.4 Offline reconstruction and selections

The offline process included: further $\tau$ identification with tighter cuts: jet calibration and cuts; $E_T^{miss}$ calibration and cuts; and Higgs boson reconstruction. These cuts were optimised for the greatest $5\sigma$ discovery reach in the $\tan\beta - m_A$ plane for each of the 3 $m_A$ scenarios.

[1]This analysis uses the $E_T$ jet recombination scheme that results in massless jets where $p_T$ is equivalent to $E_T$.

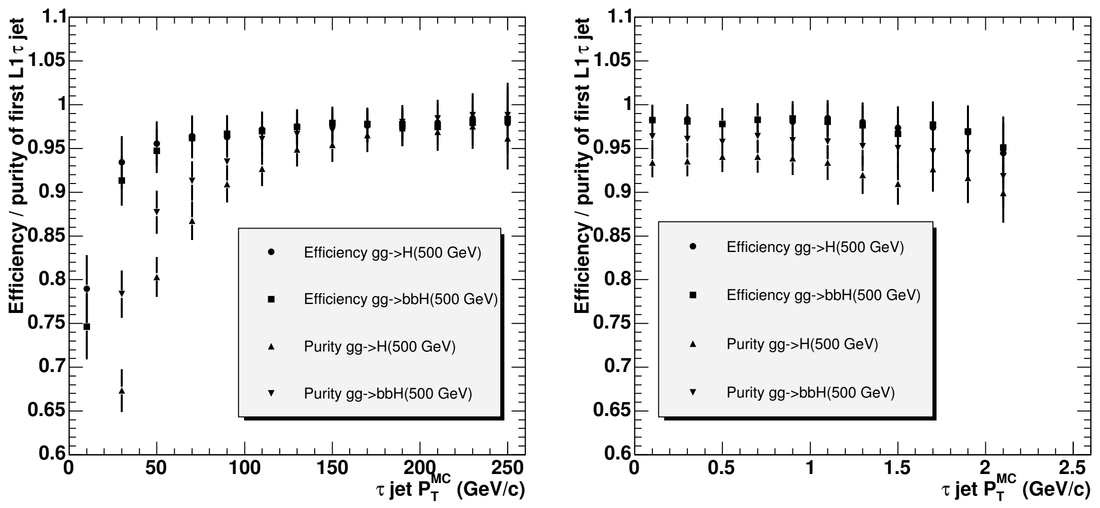


Figure 8.5: Level-1 efficiency (ratio of Level-1 to Monte carlo level $\tau$s) and purity of the most energetic $\tau$. A $\tau$ is defined as pure if it is within a distance of 0.4 in R of the true jet.

## 8.4.1 $\tau$ Identification and reconstruction

The two HLT $\tau$ candidates were reconstructed with the iterative cone algorithm [9] and cone size 0.4. The algorithm took as input calorimeter towers with thresholds of $E_T = 0.5$ GeV and $E = 0.8$ GeV with a 1 GeV seed threshold. These cuts had been optimised to suppress fake jet contamination [83]. Global track reconstruction was performed with the full tracker and parameters listed in Section 7.2.1.

The following offline $\tau$ id cuts were applied: jet $E_T$, tracker isolation and electron rejection.

### $\tau$ jet threshold

$\tau$ jet $E_T$ thresholds were used to reduce the backgrounds. Figure 8.6 shows the signal $E_T$ distribution. The difference between the $E_T$ of the two $\tau$ jets motivated the use of asymmetric $E_T$ cuts. The least energetic $\tau$ was required to have $E_T > 50$ GeV due to the background preselection. The most energetic $\tau$ $E_T$ cut was set to 65, 100 and 130 GeV for the three $m_A$ scenarios.

### Tracker isolation

Tracks reconstructed with the full tracker were used to recalculate the tracker isolation. The parameters were the same as those at the HLT, with the exceptions $p_T^{ltr} > 32$ GeV and $R_s = 0.04$. This was done to further reduce the QCD fake rate for this particular analysis. Also a requirement on the number of signal tracks was introduced. The effect of this can be seen in Figure 8.7, where the number of tracks within the signal cone around the lead track is shown for the main data samples. It can be seen that a requirement for one signal

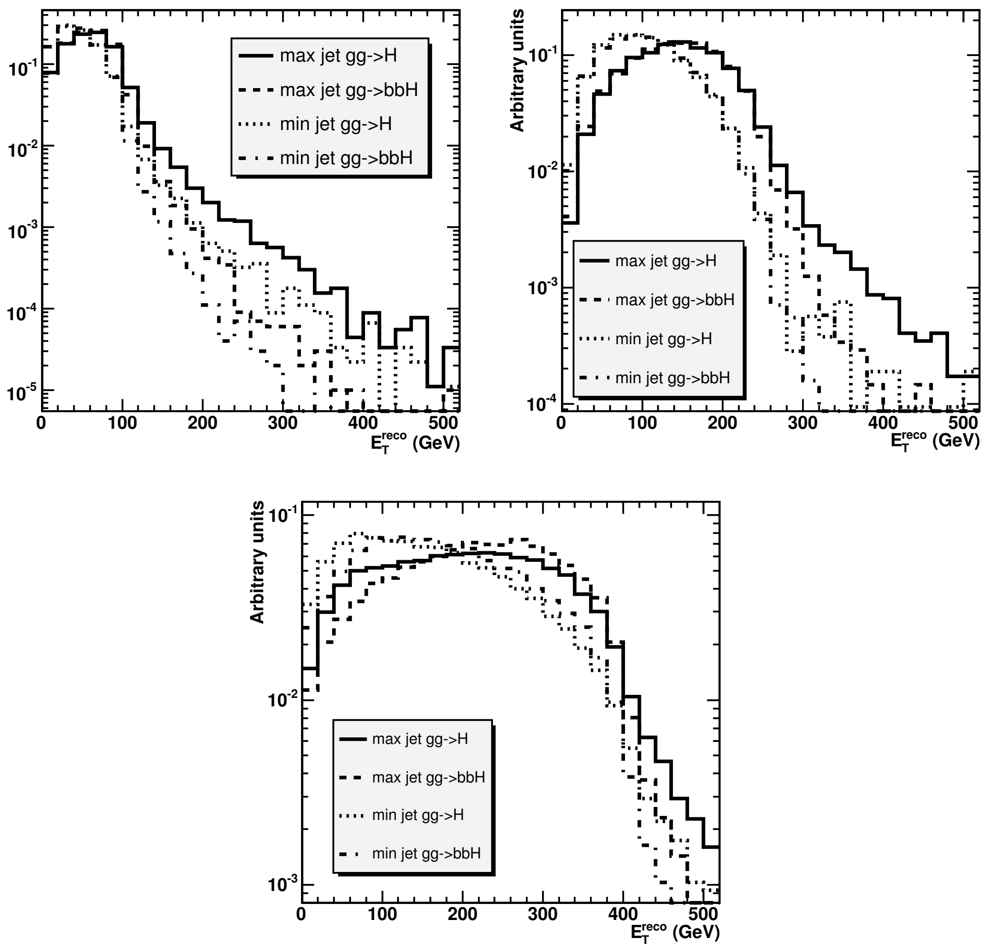


Figure 8.6: $E_T^{\rm reco}$ distribution for signal datasets, $m_A$ = 200 GeV/c$^2$ (left), $m_A$ = 500 GeV/c$^2$ (right) and $m_A$ = 800 GeV/c$^2$ (bottom).

track rejects 60–70% of the QCD multi-jet background.

### Electron rejection

To reduce Z($\gamma$) → e$^+$e$^-$ contamination a cut on the hottest HCAL tower in the jet was introduced, as described in Section 7.2.3. Both $\tau$ jet candidates were required to pass the threshold. Figure 8.8 shows the least hot HCAL tower of the two reconstructed $\tau$s in an event. A cut of 1.5 GeV was sufficient to suppress highly Z($\gamma$) → e$^+$e$^-$ .

Previous studies [10] have shown further $\tau$ tagging procedures developed at CMS based on impact parameter and the $\tau$ mass did not provide significant improvement after the trigger and offline tracker isolation.

### $\tau$ E$_T$ calibration

The $\tau$ jets were calibrated with the Monte Carlo jet E$_T$ corrections in Section 7.3.1. Figure 8.9 shows the effect of the correction. The overcorrection in the first E$_T$ bin of

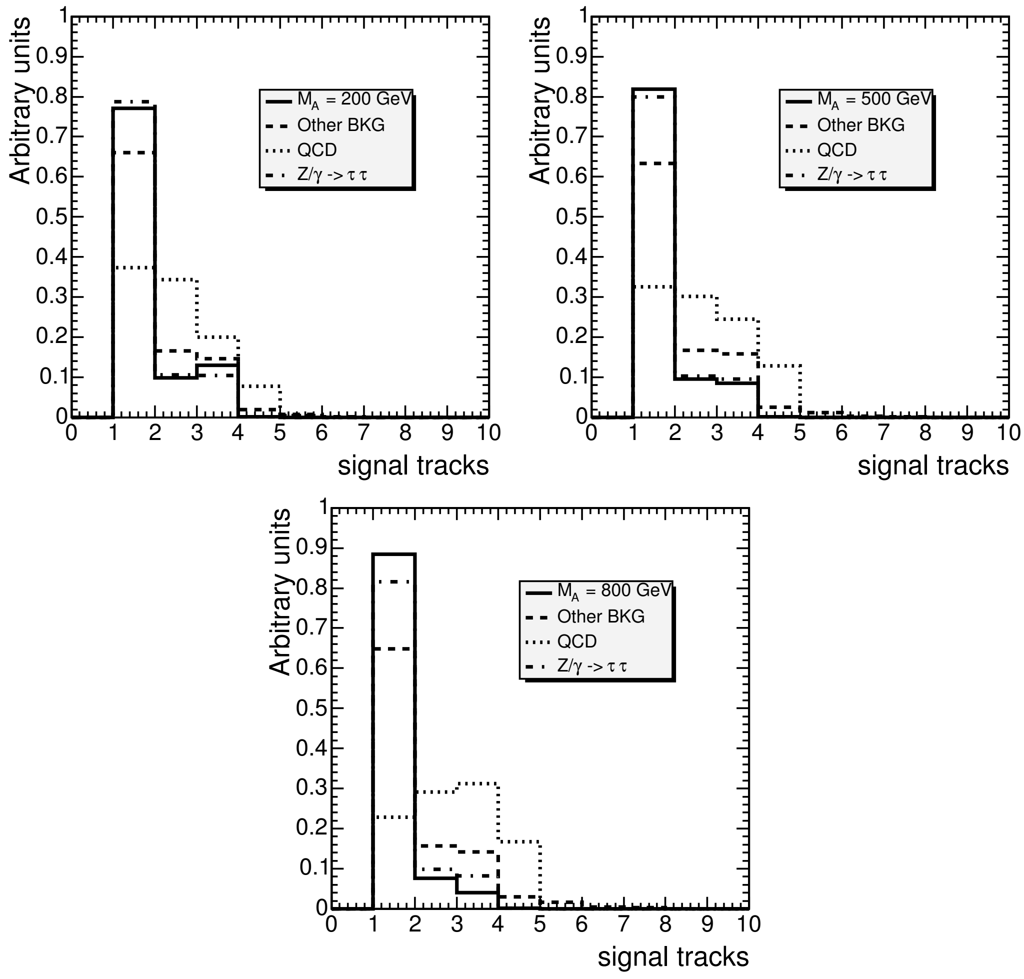


Figure 8.7: Signal track distribution for signal, QCD multi-jets, $Z(\gamma) \rightarrow \tau\tau$ and other backgrounds for $m_A = 200\,\text{GeV/c}^2$ (left), $m_A = 500\,\text{GeV/c}^2$ (right) and $m_A = 800\,\text{GeV/c}^2$ (bottom).

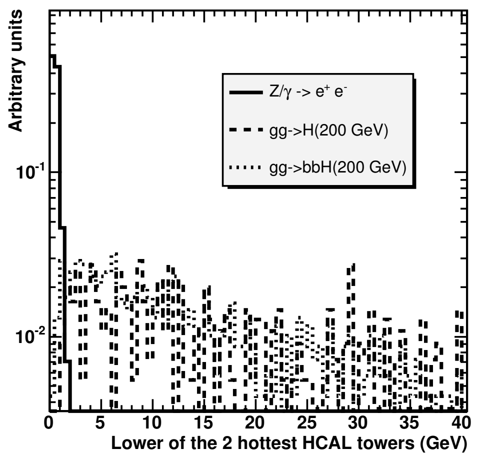


Figure 8.8: Distribution of the lowest value of the 2 $\tau$ jets hottest HCAL tower. All samples have passed both trigger levels and the offline tracker isolation.

the left plot was due to the application of tighter cuts than for the calibration. The overcorrection was not large enough to warrant a new calibration.

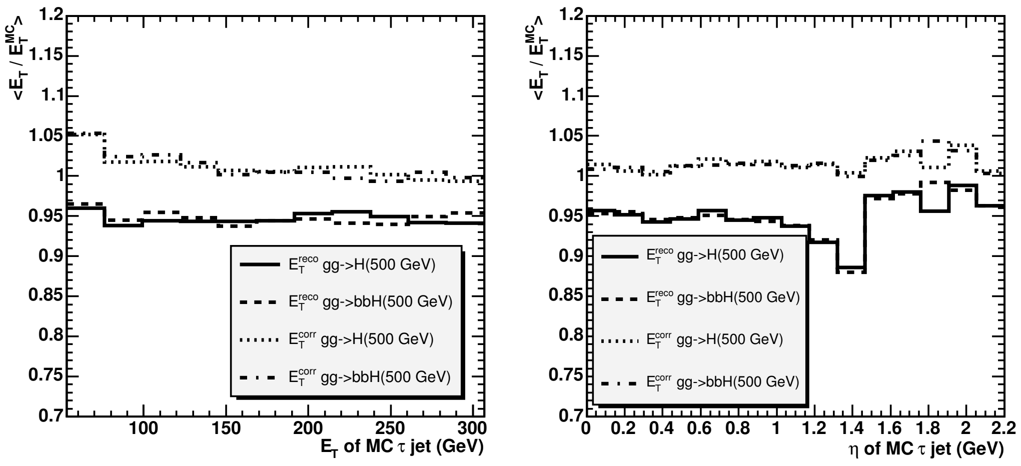


Figure 8.9: $E_T$ distribution and calibration for signal events passing cuts $\mathrm{E_T^{reco}} > 50\,\mathrm{GeV}, \mathrm{p_T^{ltr}} > 32\,\mathrm{GeV/c}$.

### 8.4.2 Missing energy

The $\mathrm{E_T^{miss}}$ for an event was measured by summing all the calorimeter towers and obtaining $\mathrm{E_x}$ and $\mathrm{E_y}$. True (MC) $\mathrm{E_T^{miss}}$ was calculated by summing all stable Monte Carlo particles except $\nu$ and $\mu$. The true $\mathrm{E_T^{miss}}$ distributions can be seen in Figure 8.10 along with the difference between the true and reconstructed $\mathrm{E_T^{miss}}$. It can be seen that the detector $\mathrm{E_T^{miss}}$ for the multi-jet background was considerably higher than the true value. This was due to limits on the detector's coverage and the imperfect calorimeter response.

Calibrations for the detector $\mathrm{E_T^{miss}}$ response had been developed [84]. These corrections involved a global jet reconstruction for jets with $\mathrm{E_T} > 25\,\mathrm{GeV}$. For each jet the constituent towers energy contributions were replaced by that of the jet. The jet energy was calibrated to correct for out-of-cone energy leakage and pileup. The correction may be written:

$$\mathrm{E_{Tx(y)}^{miss}} = -(\mathrm{E_{Tx(y)}^{raw}} + \sum_{\mathrm{jets}}(\mathrm{E_{Tx(y)}^{corr.jet}} - \mathrm{E_{Tx(y)}^{rawjet}})) \tag{8.1}$$

where $\mathrm{E_{Tx(y)}^{raw}}, \mathrm{E_{Tx(y)}^{corr.jet}}$ and $\mathrm{E_{Tx(y)}^{rawjet}}$ represent the calorimeter response, corrected and uncorrected jet energies, respectively.

Figure 8.11 shows the effect of the $\mathrm{E_T^{miss}}$ correction. For events with true (MC) $\mathrm{E_T^{miss}} < 40\,\mathrm{GeV}$ the correction can be seen to overestimate the $\mathrm{E_T^{miss}}$ of the multi-jet sample while providing a negligible correction to the Higgs events. Thus the correction was only applied to events with $\mathrm{E_T^{miss}} > 40\,\mathrm{GeV}$.

The $\mathrm{E_T^{miss}}$ cut was set to 50, 70 and 70 GeV for $m_A = 200, 500$ and $800\,\mathrm{GeV/c^2}$, respectively. The $m_A = 200\,\mathrm{GeV/c^2}$ cut had a low signal efficiency ($\sim 10\%$) but was necessary for background suppression.

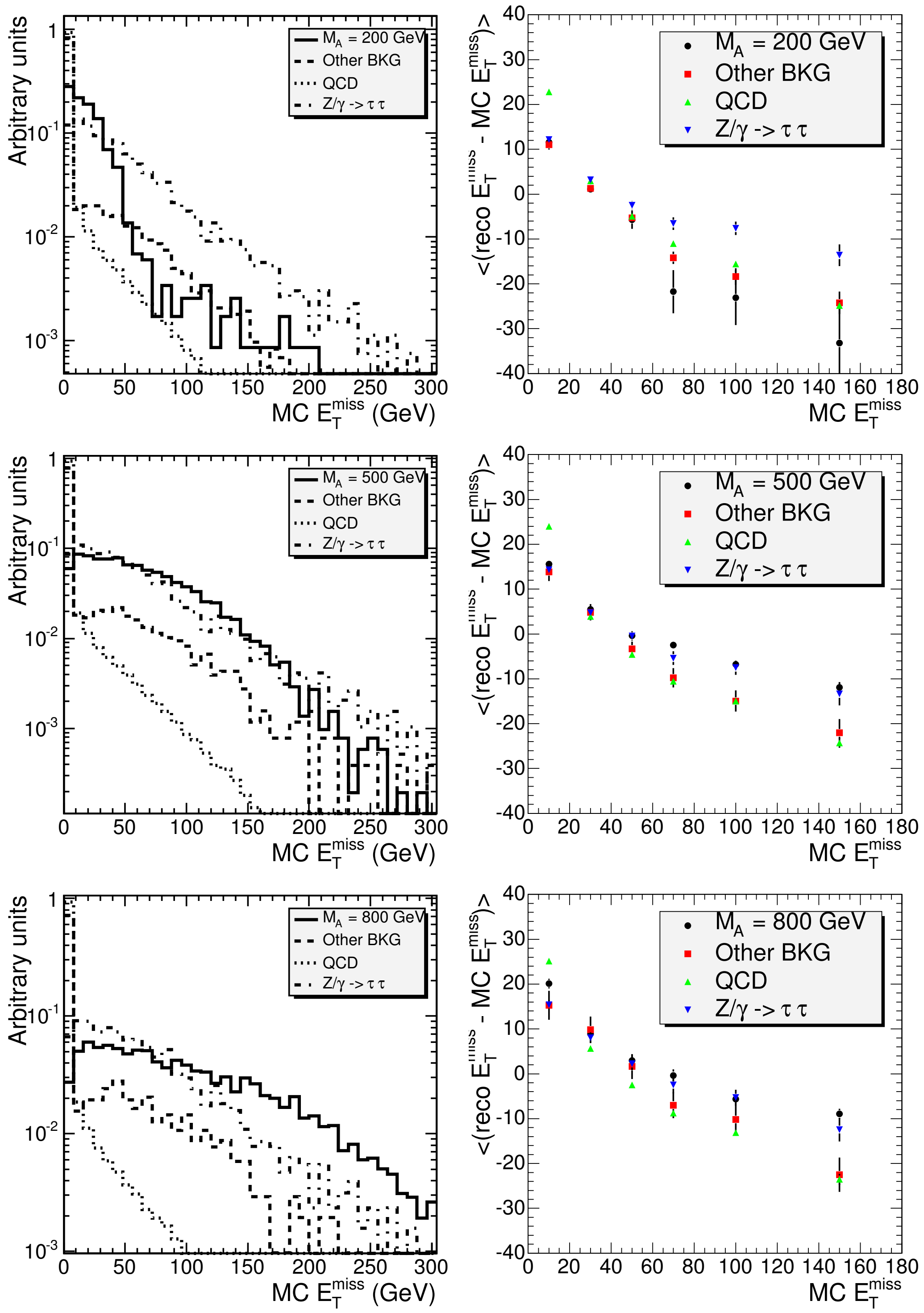


Figure 8.10: True $E_T^{miss}$ distribution (left) and difference between true and reconstructed $E_T^{miss}$ (right) for $m_A = 200\,\mathrm{GeV/c^2}$ (top), $m_A = 500\,\mathrm{GeV/c^2}$ (middle) and $m_A = 800\,\mathrm{GeV/c^2}$ (bottom). The background distributions differ between the $m_A$ plots as different cuts are applied for each scenario.

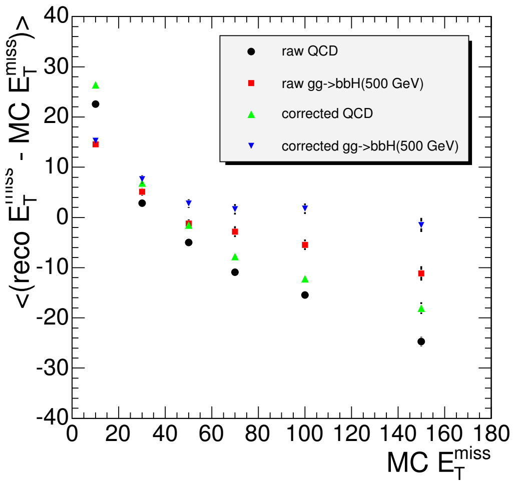


Figure 8.11: Average difference between reconstructed and true $\mathrm{E_T^{miss}}$ (GeV) in QCD and $gg \rightarrow bbH$ events, both with and without corrections applied.

### 8.4.3 Higgs mass reconstruction

The undetectable neutrinos required assumptions to be made to allow the Higgs boson mass to be reconstructed. It was assumed that the $\nu$ was emitted collinear with the $\tau$. The corrected $\mathrm{E_T^{miss}}$ could then be projected onto the two jet direction vectors. This then allowed the reconstructed Higgs mass, $\mathrm{M}_{\tau\tau}$, to be derived with:

$$\mathrm{M}_{\tau\tau} = [2\,\mathrm{E}_{\tau 1}\,\mathrm{E}_{\tau 2}(1 - \cos\theta_{\tau\tau})]^{1/2} \tag{8.2}$$

where $\mathrm{E}_{\tau 1}$, $\mathrm{E}_{\tau 2}$, $\cos\theta_{\tau\tau}$ represent the $\mathrm{E_T}$ of the two $\tau$s and the angle in $\theta$ between them.

A number of quality cuts were applied to improve the $M_{\tau\tau}$ resolution. These requirements were positive energy neutrinos and non-back-to-back $\tau$ jets. Figure 8.12 shows the $M_{\tau\tau}$ distribution as a function of the angle in $\phi$ between the jets, $\Delta\phi_{jj}$, and as each reconstruction cut was applied. The $\Delta\phi_{jj}$ cut was set to 175°.

A mass window was formed around each $\mathrm{M}_{\tau\tau}$ peak and the number of signal and background events contained within used to calculate the discovery significance. The windows used were 150–250, 400–700 and 600–1100 GeV/c$^2$ for $m_A = 200, 500$ and 800 GeV/c$^2$. Within the mass windows these cuts resulted in mass resolutions of 9.6%, 16.1% and 19.2% for $m_A$= 200, 500 and 800 GeV/c$^2$, respectively.

## 8.5 Cut overview

Table 8.2 shows a compete list of all cuts used in this analysis. The event kinematics for the different $m_A$ scenarios meant that no single set of cut parameters was appropriate for all cases. Thus the cuts were optimised separately for each of the three $m_A$ scenarios. The point of lowest 5$\sigma$ discovery in the $\tan\beta - m_A$ plane was obtained for each $m_A$ value. The

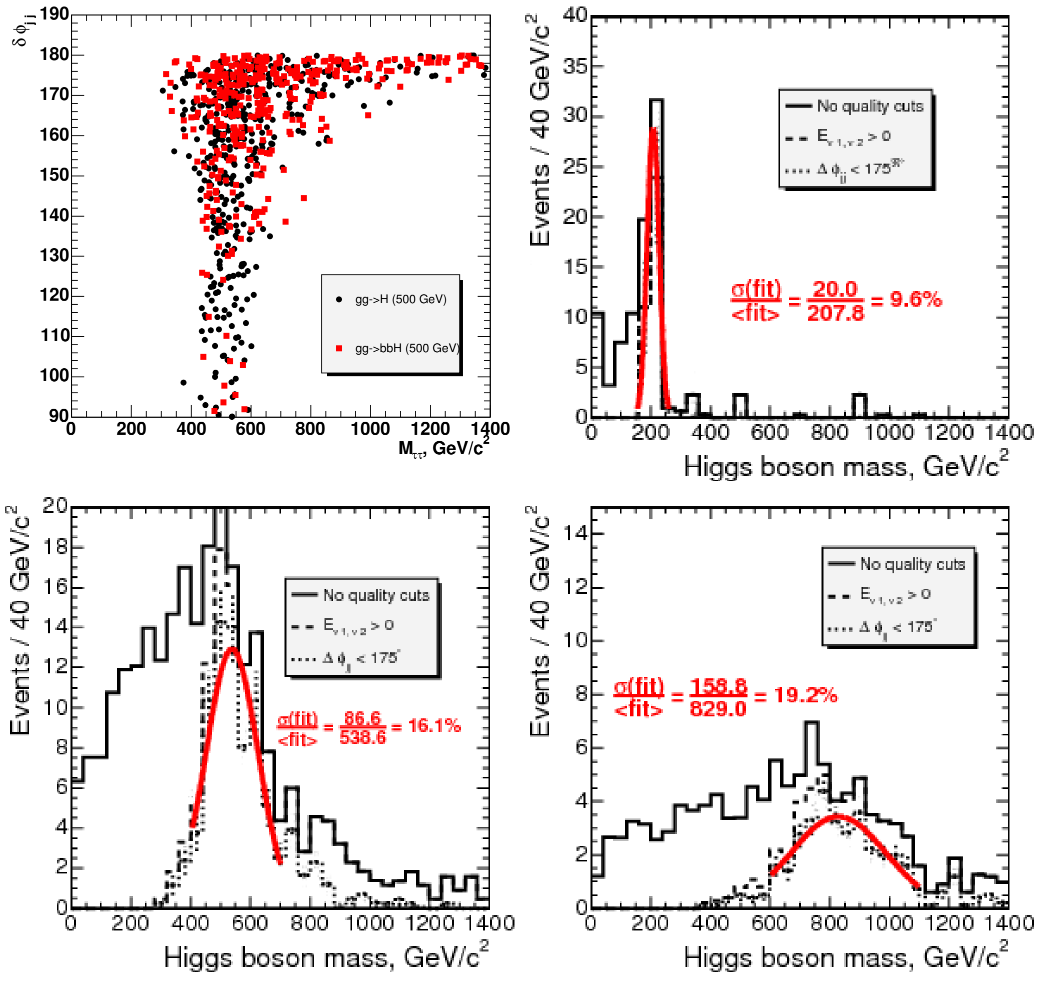


Figure 8.12: $M_{\tau\tau}$ scatter plot (top left) and distribution for $m_A = 200$ (top right), 500 (bottom left) and 800 (bottom right) GeV/c$^2$. The scatter plot is shown before the $\Delta\phi_{jj}$ cut. The distributions show M$_{\tau\tau}$ before and after the quality cuts. The mass distribution with the $\Delta\phi$ cut has been fitted with a gaussian within the mass window ranges listed in 8.4.3.

lower this number the greater the area in the $\tan\beta - m_A$ plane that could be searched. For the analysis of real data it was assumed that other searches would place limits on the allowed values of $m_A$, thus allowing the appropriate cuts to be used.

Figure 8.13 shows the optimisation of individual cuts for $\tan\beta$ reach. Cuts were not automatically set to the lowest point for every cut-$m_A$ combination due to complications introduced by the generator level selections and the need for a statistically meaningful sample of passing events. Cuts could not be set below the value used at the generator level, which imposed a lower bound on the cut. As the cuts were increased the amount of signal and background events passing all cuts sharply decreased. Thus the seemingly lowest $\tan\beta$ reach could not be relied on as it may not provide a statistically meaningful result. At least 20 events were required to pass in each dataset to allow statistically meaningful conclusions to be drawn. The final cut values taken were the lowest points in Figure 8.13 that were greater than the generator level selection whilst still statistically

| | $m_A = 200$ GeV/c$^2$ | $m_A = 500$ GeV/c$^2$ | $m_A = 800$ GeV/c$^2$ |
|---|---|---|---|
| Level-1 $\tau$ trigger | single $\tau$ $E_T = 93$ GeV or Double $\tau$ $E_T = 66$ GeV | | |
| HLT CaloPxl | $P^{cut}_{isol} = 5$ GeV, $p^{ltr}_T > 3$ GeV/c, $R_m = 0.1, R_S = 0.07, R_i = 0.5$ | | |
| 2 calorimeter jets | $E_T > 10$ GeV/c$^2$ | | |
| $E_T$ cut | 65 and 50 GeV | 100 and 50 GeV | 130 and 50 GeV |
| 2 offline $\tau$ candidates | $R_m = 0.1, p^{ltr}_T > 10$ GeV | | |
| Lead track | $p^{ltr}_T > 32$ GeV | | |
| Tracker isolation | $R_i = 0.5$ | | |
| Signal tracks | 1 track within cone $R_S = 0.04$ | | |
| $e^-$ rejection | Hottest HCAL tower $> 1.5$ GeV | | |
| Charge correlation | $Q_{\tau_1} \times Q_{\tau_1} = -1$ | | |
| $E^{miss}_T$ | $E^{miss}_T > 50$ GeV | $E^{miss}_T > 70$ GeV | $E^{miss}_T > 70$ GeV |
| Positive energy $\nu$ | $E_{\nu 1}, E_{\nu 2} > 0$ | | |
| Non-back-to-back jets | $\phi_{jj} < 175°$ | | |
| Mass window | 150–250 GeV/c$^2$ | 400–700 GeV/c$^2$ | 600–1100 GeV/c$^2$ |

Table 8.2: Summary of cuts for each $m_A$ scenario.

meaningful.

## 8.6 Efficiencies and number of events

To obtain estimates for the expected number of events a set of MSSM parameters had to be chosen. The $m_h^{max}$ scenario was used with the paramaters $M_t = 175$ GeV/c$^2, M_2 = 200$ GeV/c$^2, \mu = 200$ GeV/c$^2, X_t = 2000$ TeV/c$^2, M_{susy} = 1$ TeV/c$^2$. These parameters are suggested in Ref [78]. These were used with the FeynHiggs [85] program to obtain cross-sections. Values of $\tan\beta$ were set to be close to the expected $5\sigma$ discovery significance for 60fb$^{-1}$ of data for each of the three $m_A$ scenarios. The actual $5\sigma$ discovery significance is calculated later. The values of $\tan\beta$ used were 20, 30 and 40 for $m_A$ = 200, 500 and 800 GeV/c$^2$, respectively.

The signal and background efficiencies for all datasets except the QCD multijets and $Z(\gamma) \rightarrow e^+e^-$ for each $m_A$ scenario are shown in Tables 8.3, 8.5 and 8.7. It was not possible to apply sensible cuts and have a significant number of multi-jet and $Z(\gamma) \rightarrow e^+e^-$ events pass all cuts. Thus for these samples the cuts were factorised into three groups, where the first was applied followed independently by the second and third groups. The group efficiencies were combined to form the total efficiency. The first group contained the Level-1 trigger and offline calorimeter reconstruction. The second contained the HLT and offline $\tau$ identification. The third group contained the $E^{miss}_T$ and $M_{\tau\tau}$ quality cuts. Due to the low numbers of QCD events the requirement for oppositely charged jets was not enforced for these events. Previous studies [80] have shown that this results in a 50% efficiency, so this value was used. Efficiencies for the factorised cuts are shown in Tables 8.4, 8.6 and 8.8 for $m_A = 200, 500$ and $800$ GeV/c$^2$, respectively.

The cut factorisation was not applied to all samples as the cut groups were not independent for samples containing genuine $\tau$ jets. The $M_{\tau\tau}$ distribution obtained with relaxed

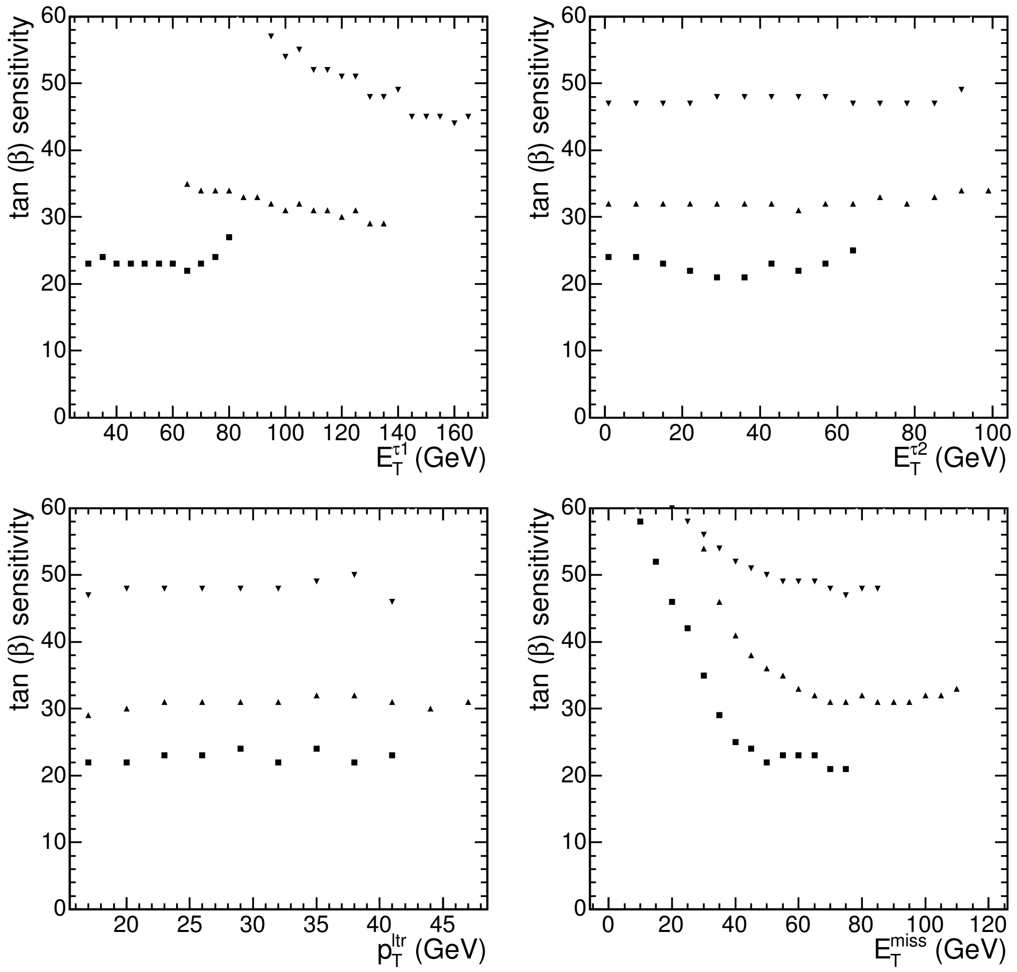


Figure 8.13: $\tan\beta$ reach as a function of cut parameters. Varied cuts are $E_T$ of most energetic $\tau$ ($E_T^{\tau1}$), $E_T$ of least energetic $\tau$ ($E_T^{\tau1}$), $p_T$ of $\tau$ lead track ($p_T^{ltr}$) and $E_T^{miss}$. Each cut is varied for 3 mass scenarios $m_A = 200$ (squares), 500 (upward pointing triangles) and 800 (downward pointing triangles) GeV/c$^2$.

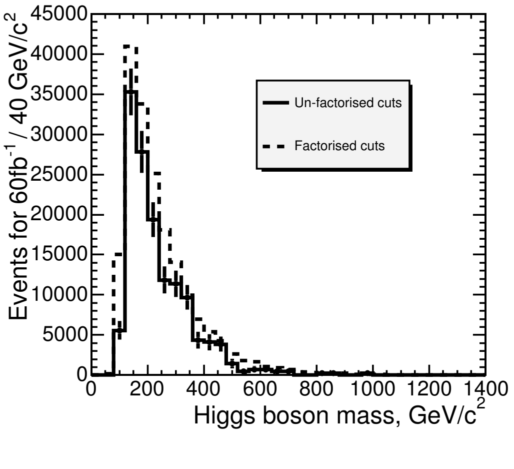


Figure 8.14: $M_{\tau\tau}$ distributions for $Z(\gamma) \rightarrow e^+e^-$ with and without factorised cut groups. Obtained with relaxed cuts: $E_T > 30$ GeV, $p_T^{ltr} > 10$ GeV, $E_T^{miss} > 10$ GeV and hot HCAL tower $> 0.5$ GeV.

cuts for $\mathrm{Z}(\gamma) \rightarrow \mathrm{e}^{+}\mathrm{e}^{-}$, with and without factorised cut groups, is shown in Figure 8.14. The two distributions were compared with the Kolmogorov test and a similarity of 27.8% obtained. It can be seen that the factorised cuts resulted in an over-estimation of the distribution. As the factorisation was only applied to background datasets this resulted in a higher background than would otherwise be expected. Thus results obtained represented a worst case scenario. A comparison of the multi-jets was not performed as not enough Monte Carlo events passed all cuts even when they were relaxed.

Some data samples that had been generated in multiple $\mathrm{E_T}$/mass ranges had no events in a set range that passed all cuts. This was because they fell outside of the signal region, i.e. their generated $\mathrm{E_T}$ was below the cut or their invariant mass was outside the mass window.

The $\mathrm{t\bar{t}}$, $\mathrm{W+jet}$ and some QCD multi-jet datasets ran out of events before passing all selections in certain $m_A$ scenarios. Factorised cuts could not be used for the $\mathrm{W+jet}$ or $\mathrm{t\bar{t}}$ samples as they contained genuine $\tau$ jets. In these cases the cuts were loosened until a significant number of Monte Carlo events passed. These were then used to obtain a prediction for the upper bound on the expected number of background events.

Figure 8.15 shows the $\mathrm{M}_{\tau\tau}$ plots for all $m_A$ scenarios. The expected number of signal and background events and the discovery significance can be seen in Table 8.9. It can be seen that the major background processes were QCD multi-jets and $\mathrm{Z}(\gamma) \rightarrow \tau\tau$.

## 8.7 Evaluation of background from data and systematics

The background uncertainty was calculated for the two main backgrounds, QCD multi-jets and $\mathrm{Z}(\gamma) \rightarrow \tau\tau$.

### 8.7.1 QCD multi-jets

The QCD multi-jet background could be directly measured from the data by replacing the requirement for jets of opposite electric charge with one for same sign jets. This requirement effectively suppressed all other signal and backgrounds. The number of same sign and opposite sign QCD multi-jet events was equal so a direct measurement of the uncertainty on the number of events was possible. Expected number of events was calculated with all cuts except the mass window. The expected events were 296.5, 34.0, and 8.6 for $m_A = 200$, 500 and 800 GeV/c$^2$ respectively, giving uncertainties of 5.8, 17.2 and 34.2%.

### 8.7.2 $\mathrm{Z}(\gamma) \rightarrow \tau\tau$

It was not possible to isolate the $\mathrm{Z}(\gamma) \rightarrow \tau\tau$ background hence the uncertainty on the number of events had to be computed. The total uncertainty, $\Delta_{\mathrm{Z}_{\tau\tau}}$, could be computed:

$$\Delta_{\mathrm{Z}_{\tau\tau}} = \Delta_{theory} \oplus \Delta_L \oplus \Delta_\epsilon \tag{8.3}$$

| | $gg \rightarrow bbA/H$ | $gg \rightarrow A/H$ | $t\bar{t}$ | W + jet | Wt | $Z(\gamma) \rightarrow \tau\tau$ $80 < M_Z < 130$ | $Z(\gamma) \rightarrow \tau\tau$ $130 < M_Z < 300$ | $Z(\gamma) \rightarrow \tau\tau$ $300 < M_Z$ |
|---|---|---|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | 3753 | 476 | 281 | 12398 | 9.8 | 29830 | 1182 | 201 |
| L1 $\tau$ Trigger | 0.501 | 0.519 | 0.753 | 0.784 | 0.826 | 0.530 | 0.702 | 0.859 |
| HLT CaloPxl | 0.291 | 0.246 | 0.047 | 0.034 | 0.128 | 0.046 | 0.192 | 0.241 |
| 2 offline calorimeter $\tau$ jets | 0.997 | 0.998 | 0.994 | 0.995 | 0.997 | 0.984 | 0.999 | 0.999 |
| $\tau E_T^{cut}$ | 0.347 | 0.348 | 0.562 | 0.725 | 0.585 | 0.241 | 0.415 | 0.842 |
| 2 offline $\tau$ candidates | 0.678 | 0.688 | 0.815 | 0.800 | 0.795 | 0.693 | 0.699 | 0.713 |
| $p_T^{ltr}$ cut | 0.404 | 0.403 | 0.661 | 0.639 | 0.749 | 0.290 | 0.410 | 0.600 |
| Tracker isolation | 0.855 | 0.854 | 0.825 | 0.476 | 0.852 | 0.638 | 0.850 | 0.918 |
| Signal tracks | 0.592 | 0.587 | 0.567 | 0.254 | 0.544 | 0.378 | 0.580 | 0.643 |
| Hottest HCAL Tower | 0.954 | 0.989 | 0.903 | 0.965 | 0.921 | 0.941 | 0.904 | 0.877 |
| $Q_{\tau 1}xQ_{\tau 2}$ = -1 | 0.979 | 0.962 | 0.958 | 0.835 | 0.940 | 0.594 | 0.950 | 0.943 |
| $E_T^{miss}$ | 0.088 | 0.128 | 0.607 | 0.295 | 0.472 | 0.158 | 0.129 | 0.443 |
| $E_{\nu 1}, E_{\nu 2} > 0$ | 0.298 | 0.507 | 0.426 | 0.138 | 0.383 | 0.333 | 0.475 | 0.470 |
| $\phi_{jj} < 175°$ | 0.824 | 0.882 | 0.867 | 0.467 | 0.861 | 1.000 | 0.679 | 0.502 |
| Mass window efficiencies | $1.21 \times 10^{-4}$ | $2.77 \times 10^{-4}$ | 0.000 | 0.000 | $6.67 \times 10^{-5}$ | 0.000 | $5.9 \times 10^{-5}$ | 0.000 |
| Mass window $\sigma$(fb) | 0.454 | 0.133 | 0.000 | 0.000 | $6.53\times10^{-4}$ | 0.000 | 0.0695 | 0.000 |
| expected events for 60 (fb$^{-1}$) | 27.3 | 7.94 | 0.000 | 0.000 | 0.0391 | 0.000 | 4.17 | 0.000 |

Table 8.3: Efficiencies for signal and background events in the $m_A = 200\,\text{GeV}$ scenario. $M_Z$ in GeV/c$^2$

| | QCD multi − jets $50 < \hat{p}_T < 80$ GeV | QCD multi − jets $80 < \hat{p}_T < 120$ GeV | QCD multi − jets $120 < \hat{p}_T < 170$ GeV | QCD multi − jets $\hat{p}_T > 170$ GeV | $Z(\gamma) \to e^+e^-$ |
|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | $5.08 \times 10^6$ | $1.70 \times 10^7$ | $2.11 \times 10^7$ | $2.82 \times 10^7$ | $3.97 \times 10^6$ |
| L1 $\tau$ Trigger | 0.461 | 0.716 | 0.727 | 0.562 | 0.106 |
| 2 offline calorimeter $\tau$ jets | 0.995 | 0.982 | 0.976 | 0.976 | 0.995 |
| $\tau E_T^{cut}$ | 0.121 | 0.665 | 0.784 | 0.742 | 0.060 |
| Group 1 cuts | 0.0551 | 0.462 | 0.544 | 0.393 | 0.00622 |
| HLT CaloPxl | 0.012 | 0.005 | 0.002 | 0.001 | 0.133 |
| 2 offline $\tau$ candidates | 0.879 | 0.839 | 0.860 | 0.830 | 0.554 |
| $p_T^{ltr}$ cut | 0.608 | 0.647 | 0.569 | 0.547 | 0.741 |
| Tracker isolation | 0.323 | 0.244 | 0.264 | 0.216 | 0.907 |
| Signal tracks | 0.100 | 0.216 | 0.250 | 0.067 | 0.937 |
| Hottest HCAL Tower | 1.000 | 1.000 | 1.000 | 1.000 | 0.007 |
| $Q_{\tau 1}$x$Q_{\tau 2}$ = -1 | 0.500 | 0.500 | 0.500 | 0.500 | 1.00 |
| Group 2 cuts | $2.00\times10^{-3}$ | $1.32\times10^{-3}$ | $5.75 \times10^{-5}$ | $5.21 \times10^{6}$ | $3.26\times10^{-3}$ |
| $E_T^{miss}$ | 0.006 | 0.008 | 0.024 | 0.086 | 0.047 |
| $E_{\nu 1}, E_{\nu 2} > 0$ | 0.448 | 0.474 | 0.430 | 0.482 | 0.502 |
| $\phi_{jj} < 175°$ | 0.462 | 0.720 | 0.684 | 0.741 | 0.842 |
| Group 3 cuts | 0.00120 | 0.00264 | 0.00712 | 0.0308 | 0.0201 |
| Mass window efficiencies | 0.000 | $7.75 \times10^{-9}$ | $8.26 \times10^{-9}$ | $3.34 \times10^{-9}$ | $1.99 \times10^{-9}$ |
| Mass window $\sigma$(fb) | 0.000 | 0.132 | 0.174 | 0.0943 | 0.00789 |
| expected events for 60 (fb$^{-1}$) | 0.000 | 7.89 | 10.4 | 5.66 | 0.473 |

Table 8.4: Efficiencies for factorised background events in the $m_A = 200$ GeV scenario

| | gg $\rightarrow$ bbA/H | gg $\rightarrow$ A/H | $t\bar{t}$ | W + jet | Wt | Z($\gamma$) $\rightarrow \tau\tau$ 80 < $M_Z$ < 130 | Z($\gamma$) $\rightarrow \tau\tau$ 130 < $M_Z$ < 300 | Z($\gamma$) $\rightarrow \tau\tau$ 300 < $M_Z$ |
|---|---|---|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | 186 | 6.31 | 281 | 12398 | 9.80 | 29830 | 1181 | 201 |
| L1 $\tau$ Trigger | 0.853 | 0.858 | 0.753 | 0.784 | 0.826 | 0.530 | 0.702 | 0.859 |
| HLT CaloPxl | 0.322 | 0.267 | 0.047 | 0.034 | 0.128 | 0.046 | 0.192 | 0.241 |
| 2 offline calorimeter $\tau$ jets | 0.999 | 0.999 | 0.994 | 0.995 | 0.997 | 0.984 | 0.999 | 0.999 |
| $\tau E_T^{cut}$ | 0.756 | 0.753 | 0.270 | 0.349 | 0.287 | 0.073 | 0.097 | 0.639 |
| 2 offline $\tau$ candidates | 0.717 | 0.710 | 0.816 | 0.812 | 0.778 | 0.737 | 0.725 | 0.717 |
| $p_T^{ltr}$ cut | 0.662 | 0.659 | 0.671 | 0.667 | 0.776 | 0.459 | 0.505 | 0.634 |
| Tracker isolation | 0.950 | 0.942 | 0.845 | 0.486 | 0.852 | 0.583 | 0.840 | 0.922 |
| Signal tracks | 0.671 | 0.662 | 0.570 | 0.211 | 0.538 | 0.286 | 0.562 | 0.653 |
| Hottest HCAL Tower | 0.984 | 0.976 | 0.904 | 0.945 | 0.927 | 1.000 | 0.902 | 0.881 |
| $Q_{\tau1}$x$Q_{\tau2}$ = -1 | 0.937 | 0.947 | 0.935 | 0.824 | 0.922 | 0.583 | 0.942 | 0.941 |
| $E_T^{miss}$ | 0.385 | 0.436 | 0.462 | 0.290 | 0.406 | 0.143 | 0.115 | 0.327 |
| $E_{\nu1}, E_{\nu2} > 0$ | 0.428 | 0.529 | 0.452 | 0.133 | 0.372 | 0.000 | 0.333 | 0.477 |
| $\phi_{jj} < 175°$ | 0.523 | 0.753 | 0.848 | 0.500 | 0.812 | 0.000 | 0.800 | 0.527 |
| Mass window efficiencies | 0.004 | $6.76\times10^{-4}$ | $1.15\times10^{-4}$ | $5.04\times10^{-6}$ | $5.33\times10^{-4}$ | 0.000 | 0.000 | 0.00125 |
| Mass window $\sigma$(fb) | 0.745 | 0.0426 | 0.0324 | 0.0625 | 0.00523 | 0.000 | 0.000 | 0.252 |
| expected events for 60 (fb$^{-1}$) | 44.7 | 2.56 | 1.95 | 3.75 | 0.313 | 0.000 | 0.000 | 15.1 |

Table 8.5: Efficiencies for signal and background events in the $m_A = 500\,\text{GeV}$ scenario. $M_Z$ in GeV/c$^2$

| | QCD multi − jets $50 < \hat{p}_T < 80$ GeV | QCD multi − jets $80 < \hat{p}_T < 120$ GeV | QCD multi − jets $120 < \hat{p}_T < 170$ GeV | QCD multi − jets $\hat{p}_T > 170$ GeV | $Z(\gamma) \rightarrow e^+e^-$ |
|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | $5.08 \times 10^6$ | $1.70 \times 10^7$ | $2.11 \times 10^7$ | $2.82 \times 10^7$ | $3.97 \times 10^6$ |
| L1 $\tau$ Trigger | 0.461 | 0.716 | 0.727 | 0.562 | 0.106 |
| 2 offline calorimeter $\tau$ jets | 0.995 | 0.982 | 0.976 | 0.976 | 0.995 |
| $\tau E_T^{cut}$ | 0.001 | 0.061 | 0.467 | 0.667 | 0.020 |
| Group 1 cuts | $4.61\times10^{-4}$ | 0.0421 | 0.324 | 0.354 | 0.00205 |
| HLT CaloPxl | 0.024 | 0.004 | 0.002 | 0.001 | 0.112 |
| 2 offline $\tau$ candidates | 1.000 | 0.903 | 0.848 | 0.838 | 0.623 |
| $p_T^{ltr}$ cut | 0.000 | 0.679 | 0.571 | 0.564 | 0.768 |
| Tracker isolation | 0.000 | 0.158 | 0.234 | 0.195 | 0.890 |
| Signal tracks | 0.000 | 0.667 | 0.200 | 0.040 | 0.897 |
| Hottest HCAL Tower | 0.000 | 1.000 | 1.000 | 1.000 | 0.011 |
| $Q_{\tau1}$x$Q_{\tau2}$ = -1 | 0.000 | 0.500 | 0.500 | 0.500 | 1.00 |
| Group 2 cuts | 0.000 | $2.65\times10^{-4}$ | $4.83 \times10^{-5}$ | $2.90 \times10^{-6}$ | $4.93\times10^{-3}$ |
| $E_T^{miss}$ | 0.024 | 0.001 | 0.004 | 0.034 | 0.022 |
| $E_{\nu1}, E_{\nu2} > 0$ | 1.000 | 0.556 | 0.535 | 0.555 | 0.523 |
| $\phi_{jj} < 175°$ | 0.000 | 0.800 | 0.809 | 0.819 | 0.783 |
| Group 3 cuts | 0.000 | $5.29\times10^{-4}$ | 0.00177 | 0.0155 | 0.00888 |
| Mass window efficiencies | 0.000 | $7.37 \times10^{-10}$ | $6.81 \times10^{-9}$ | $2.61 \times10^{-9}$ | $5.01 \times10^{-9}$ |
| Mass window $\sigma$(fb) | 0.000 | 0.0125 | 0.144 | 0.0736 | 0.0199 |
| expected events for 60 (fb$^{-1}$) | 0.000 | 0.750 | 8.61 | 4.42 | 1.19 |

Table 8.6: Efficiencies for factorised background events in the $m_A = 500$ GeV scenario

| | gg → bbA/H | gg → A/H | $t\bar{t}$ | W + jet | Wt | $Z(\gamma) \to \tau\tau$ $80 < M_Z < 130$ | $Z(\gamma) \to \tau\tau$ $130 < M_Z < 300$ | $Z(\gamma) \to \tau\tau$ $300 < M_Z$ |
|---|---|---|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | 49 | 0.722 | 281 | 12398 | 9.80 | 29830 | 1181 | 200 |
| L1 $\tau$ Trigger | 0.896 | 0.875 | 0.753 | 0.784 | 0.826 | 0.530 | 0.702 | 0.859 |
| HLT CaloPxl | 0.316 | 0.247 | 0.047 | 0.034 | 0.128 | 0.046 | 0.192 | 0.241 |
| 2 offline calorimeter $\tau$ jets | 1.000 | 0.999 | 0.994 | 0.995 | 0.997 | 0.984 | 0.999 | 0.999 |
| $\tau E_T^{cut}$ | 0.832 | 0.782 | 0.121 | 0.170 | 0.132 | 0.027 | 0.019 | 0.411 |
| 2 offline $\tau$ candidates | 0.679 | 0.679 | 0.796 | 0.813 | 0.787 | 0.722 | 0.762 | 0.720 |
| $p_T^{ltr}$ cut | 0.746 | 0.743 | 0.688 | 0.694 | 0.782 | 0.421 | 0.557 | 0.663 |
| Tracker isolation | 0.951 | 0.948 | 0.845 | 0.500 | 0.858 | 0.667 | 0.795 | 0.930 |
| Signal tracks | 0.785 | 0.774 | 0.653 | 0.185 | 0.661 | 0.062 | 0.586 | 0.675 |
| Hottest HCAL Tower | 0.983 | 0.984 | 0.948 | 0.967 | 0.917 | 1.000 | 0.882 | 0.874 |
| $Q_{\tau 1}xQ_{\tau 2}$ = -1 | 0.936 | 0.941 | 0.945 | 0.795 | 0.924 | 0.000 | 0.900 | 0.940 |
| $E_T^{miss}$ | 0.568 | 0.601 | 0.477 | 0.543 | 0.475 | 0.000 | 0.407 | 0.400 |
| $E_{\nu 1}, E_{\nu 2} > 0$ | 0.468 | 0.553 | 0.366 | 0.132 | 0.328 | 0.000 | 0.364 | 0.450 |
| $\phi_{jj} < 175°$ | 0.452 | 0.664 | 0.800 | 0.400 | 0.789 | 0.000 | 0.750 | 0.498 |
| Mass window efficiencies | 0.00827 | 0.0108 | $8.97\times10^{-5}$ | $2.52\times10^{-6}$ | $2.66\times10^{-4}$ | 0.000 | 0.000 | $7.00\times10^{-4}$ |
| Mass window $\sigma$(fb) | 0.405 | 0.00783 | 0.0252 | 0.0312 | 0.00261 | 0.000 | 0.000 | 0.141 |
| expected events for 60 ($fb^{-1}$) | 24.3 | 0.470 | 1.51 | 1.87 | 0.157 | 0.000 | 0.000 | 8.44 |

Table 8.7: Efficiencies for signal and background events in the $m_A = 800\,\text{GeV}$ scenario. $M_Z$ in GeV/c$^2$

| | QCD multi − jets $50 < \hat{p}_T < 80$ GeV | QCD multi − jets $80 < \hat{p}_T < 120$ GeV | QCD multi − jets $120 < \hat{p}_T < 170$ GeV | QCD multi − jets $\hat{p}_T > 170$ GeV | $Z(\gamma) \rightarrow e^+e^-$ |
|---|---|---|---|---|---|
| $\sigma \times BR \times \epsilon_{kine}$ | $5.08 \times 10^6$ | $1.70 \times 10^7$ | $2.11 \times 10^7$ | $2.82 \times 10^7$ | $3.97 \times 10^6$ |
| L1 $\tau$ Trigger | 0.461 | 0.716 | 0.727 | 0.562 | 0.106 |
| 2 offline calorimeter $\tau$ jets | 0.995 | 0.982 | 0.976 | 0.976 | 0.995 |
| $\tau E_T^{cut}$ | $9.68 \times 10^{-5}$ | 0.003 | 0.099 | 0.536 | 0.007 |
| Group 1 cuts | $4.39 \times 10^{-5}$ | 0.00231 | 0.0689 | 0.285 | 0.000765 |
| HLT CaloPxl | 0.000 | 0.007 | 0.002 | 0.001 | 0.101 |
| 2 offline $\tau$ candidates | 0.000 | 0.667 | 0.862 | 0.811 | 0.618 |
| $p_T^{ltr}$ cut | 0.000 | 0.500 | 0.640 | 0.574 | 0.872 |
| Tracker isolation | 0.000 | 0.000 | 0.125 | 0.218 | 0.902 |
| Signal tracks | 0.000 | 0.000 | 0.000 | 0.045 | 0.865 |
| Hottest HCAL Tower | 0.000 | 0.000 | 0.000 | 1.000 | 0.031 |
| $Q_{\tau 1}$x$Q_{\tau 2}$ = -1 | 0.000 | 0.500 | 0.500 | 0.500 | 1.00 |
| Group 2 cuts | 0.000 | 0.000 | 0.000 | $3.61 \times 10^{-6}$ | 0.00133 |
| $E_T^{miss}$ | 0.000 | 0.000 | 0.008 | 0.033 | 0.038 |
| $E_{\nu 1}, E_{\nu 2} > 0$ | 0.000 | 0.000 | 0.535 | 0.542 | 0.483 |
| $\phi_{jj} < 175°$ | 0.000 | 0.000 | 0.759 | 0.786 | 0.714 |
| Group3 cuts | 0.000 | 0.000 | 0.00311 | 0.0139 | 0.0133 |
| Mass window efficiencies | 0.000 | 0.000 | 0.000 | $2.35 \times 10^{-9}$ | $6.73 \times 10^{-9}$ |
| Mass window $\sigma$(fb) | 0.000 | 0.000 | 0.000 | 0.0661 | 0.0267 |
| expected events for 60 (fb$^{-1}$) | 0.000 | 0.000 | 0.000 | 3.97 | 1.60 |

Table 8.8: Efficiencies for factorised background events in the $m_A = 800$ GeV scenario

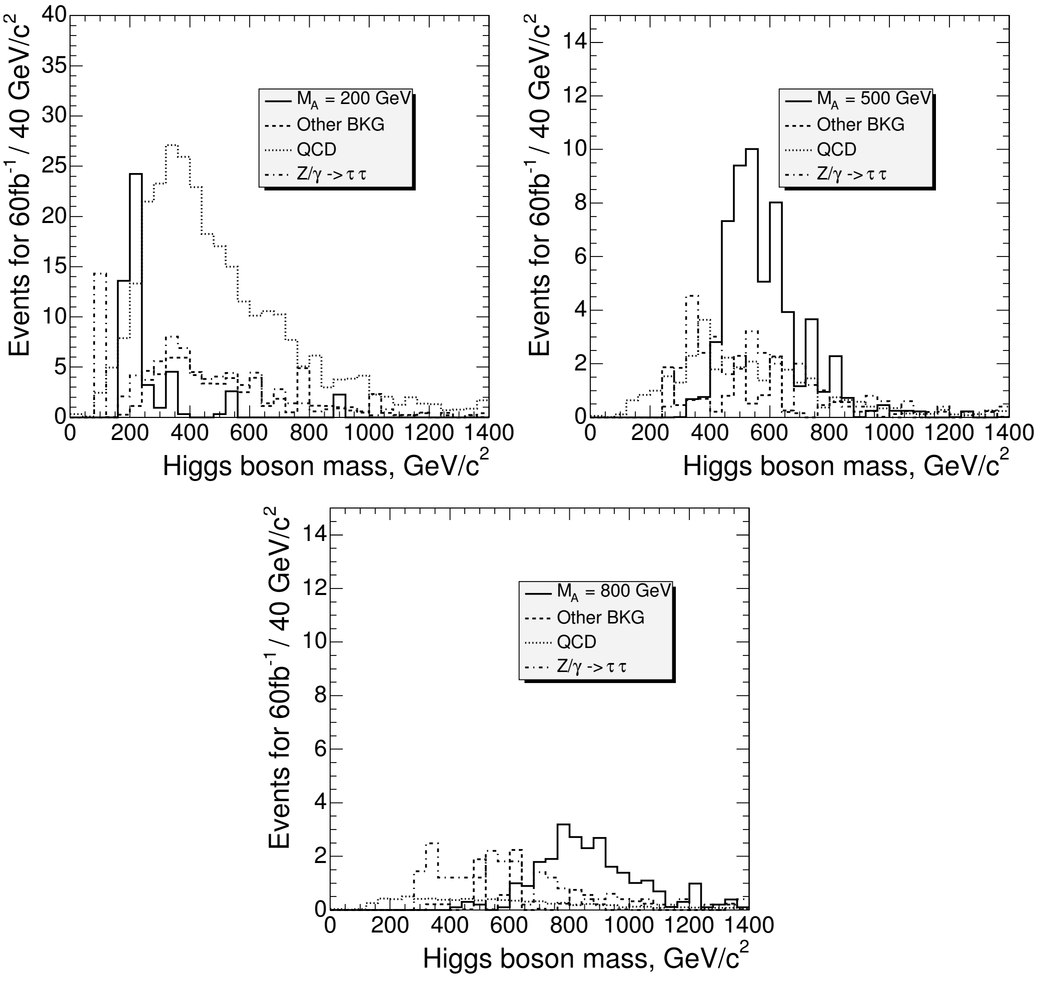


Figure 8.15: $M_{\tau\tau}$ distributions for expected signal and background for $m_A$ = 200, 500 and 800 GeV/c$^2$ in the $m_h^{max}$ scenario with tan$\beta$ = 20, 30 and 40 scaled for 60 fb$^{-1}$ of data.

| process | $m_A$ ( GeV/c$^2$) | | |
|---|---|---|---|
| | 200 | 500 | 800 |
| gg → bbA/H | 27.3 | 44.7 | 24.3 |
| gg → A/H | 7.9 | 2.6 | 0.47 |
| QCD multijets | < 25.6 | 13.8 | < 6.7 |
| Z($\gamma$) → $\tau\tau$ | 4.2 | 15.1 | 8.4 |
| Z($\gamma$) → e$^+$e$^-$ | 0.5 | 1.2 | 1.6 |
| t$\bar{\text{t}}$ | < 0.43 | 1.9 | 1.5 |
| W + jet | < 1.87 | 3.7 | 1.9 |
| Wt | 0.04 | 0.31 | 0.16 |
| $S/\sqrt{B}$ | 6.2 | 7.8 | 5.5 |

Table 8.9: Summary of expected events for all 3 mass scenarios for 60fb$^{-1}$ in the $m_h^{max}$ scenario.

with the terms on the right-hand side representing the uncertainties on the theory (cross section and branching ratio), LHC luminosity and experimental efficiency, respectively. The uncertainty on the Z → $\tau\tau$ branching ratio was small ($\simeq$ 0.1%) and hence neglected [75]. $\Delta_{theory} = 3\%$ was taken from the uncertainty on the cross section [86]. The nominal CMS value for $\Delta_L$ was 3% for 30 fb$^{-1}$ of data and above [10].

The experimental selection uncertainty could be computed:

$$\Delta_\epsilon = \Delta_{jet\,scale} \oplus \Delta_{MET} \oplus \Delta_{\tau\,tagging} \tag{8.4}$$

where the energy terms on the right-hand side represent the uncertainty of the experimental selection due to the jet scale, $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ scale and $\tau$ tagging, respectively. The $\tau$ tagging efficiency was calculated as 8.8% for each $\tau$ [87].

## Jet scale

CMS has calculated that, with 30 fb$^{-1}$ of data or more, the CMS jet scale uncertainty will be 3% and the calorimeter scale 10% [10]. No study of the $\tau$ jet scale systematics had been performed to date so the standard jet uncertainty was used for both $\tau$ and multi-jets. The effect of the jet scale uncertainty is shown in Table 8.10.

| $m_A$ scenario | $\epsilon_0$ | $\epsilon_-$ | $\epsilon_+$ | $\epsilon_+ - \epsilon_-$ | % variation |
|---|---|---|---|---|---|
| 200 GeV/c$^2$ | 0.000324 | 0.000309 | 0.000353 | 0.0000221 | 6.18 |
| 500 GeV/c$^2$ | 0.00245 | 0.00233 | 0.00258 | 0.000125 | 5.10 |
| 800 GeV/c$^2$ | 0.00167 | 0.00155 | 0.00178 | 0.000117 | 7.00 |

Table 8.10: Z($\gamma$) $\to\tau\tau$ efficiency for 3 different jet scale scenarios. Scenarios are nominal jet scale $\epsilon_0$ and $\pm 3\%$ ($\epsilon_\pm$) variation.

## $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ scale

$\Delta_{MET}$ was calculated for the corrected $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ as shown in Equation 8.1. The equation may be rewritten to separate the terms dependent on the two calorimeter scale uncertainties:

$$\mathrm{E}_{\mathrm{Tx(y)}}^{\mathrm{miss}} = -([\mathrm{E}_{\mathrm{Tx(y)}}^{\mathrm{raw}} - \sum_{\mathrm{jets}} \mathrm{E}_{\mathrm{Tx(y)}}^{\mathrm{rawjet}}]_{\mathrm{low\ E_T}} + [\sum_{\mathrm{jets}} \mathrm{E}_{\mathrm{Tx(y)}}^{\mathrm{corr.jet}}]_{\mathrm{high\ E_T}}) \tag{8.5}$$

The calorimeter and jet scale variations were applied independently and the maximal deviations measured. Results for the various $m_A$ scenarios are shown in Tables 8.11, 8.12 and 8.13.

| Calorimeter scale variation | jet scale variation | $\epsilon$ |
|---|---|---|
| 1.0 | 1.0 | 0.000324 |
| 1.1 | 1.03 | 0.000353 |
| 1.1 | 0.97 | 0.000338 |
| 0.9 | 1.03 | 0.000309 |
| 0.9 | 0.97 | 0.000294 |
| Maximal variation | | 9.09 % |

Table 8.11: Z($\gamma$) $\to\tau\tau$ $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ efficiency for various calorimeter and jet scale variations in the $m_A = 200$ GeV/c$^2$ scenario.

| Calorimeter scale variation | jet scale variation | $\epsilon$ |
|---|---|---|
| 1.0 | 1.0 | 0.00245 |
| 1.1 | 1.03 | 0.00272 |
| 1.1 | 0.97 | 0.00275 |
| 0.9 | 1.03 | 0.00232 |
| 0.9 | 0.97 | 0.00232 |
| Maximal range | | 8.84 % |

Table 8.12: $Z(\gamma) \rightarrow \tau\tau$ $E_T^{miss}$ efficiency for various calorimeter and jet scale variations in the $m_A = 500\,GeV/c^2$ scenario.

| Calorimeter scale variation | jet scale variation | $\epsilon$ |
|---|---|---|
| 1.0 | 1.0 | 0.00167 |
| 1.1 | 1.03 | 0.00183 |
| 1.1 | 0.97 | 0.00173 |
| 0.9 | 1.03 | 0.00157 |
| 0.9 | 0.97 | 0.00153 |
| Maximal range | | 9.00 % |

Table 8.13: $Z(\gamma) \rightarrow \tau\tau$ $E_T^{miss}$ efficiency for various calorimeter and jet scale variations in the $m_A = 800\,GeV/c^2$ scenario.

### Total $Z(\gamma) \rightarrow \tau\tau$ uncertainty

The $Z(\gamma) \rightarrow \tau\tau$ background uncertainty constituents were combined and the final values computed are shown in Table 8.14.

| $m_A$( GeV/c$^2$) | $\Delta_{theory}$ | $\Delta_L$ | $\Delta_{jetscale}$ | $\Delta_{MET}$ | $\Delta_{\tau tagging}$ | $\Delta_{Z_{\tau\tau}}$ |
|---|---|---|---|---|---|---|
| 200 | 0.03 | 0.03 | 0.0618 | 0.0909 | 2 × 0.088 | 0.166 |
| 500 | 0.03 | 0.03 | 0.0510 | 0.0884 | 2 × 0.088 | 0.158 |
| 800 | 0.03 | 0.03 | 0.07 | 0.09 | 2 × 0.088 | 0.166 |

Table 8.14: $Z(\gamma) \rightarrow \tau\tau$ background uncertainty for $m_A = 200, 500$ and $800\,GeV/c^2$.

### 8.7.3 Total uncertainty

The total background uncertainties were then formed from the QCD multi-jet and $Z(\gamma) \rightarrow \tau\tau$ uncertainties :

$$\Delta_{200} = 5.8 \oplus 16.6 = 17.6\% \tag{8.6}$$

$$\Delta_{500} = 17.2 \oplus 15.8 = 23.4\% \tag{8.7}$$

$$\Delta_{800} = 34.2 \oplus 16.6 = 38.0\% \tag{8.8}$$

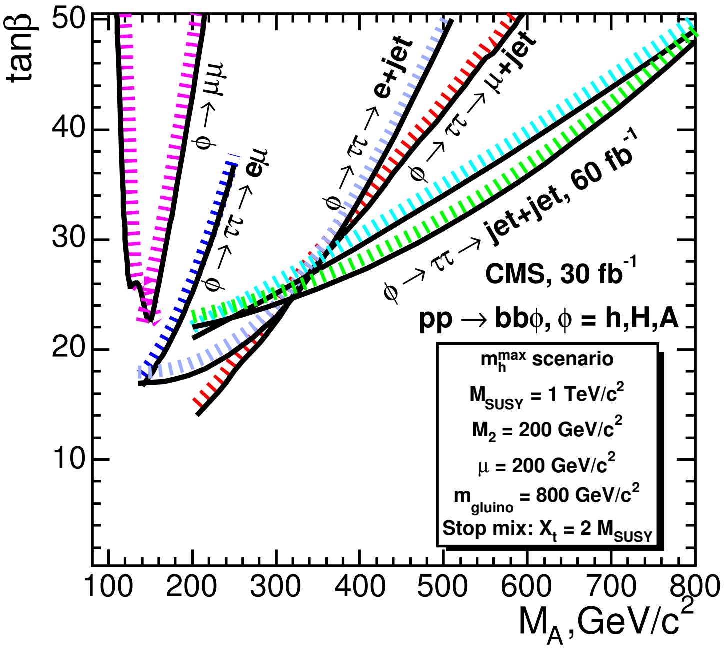


Figure 8.16: $5\sigma$ reach in the $\tan\beta - m_A$ plane in the $m_h^{max}$ scenario. Both this analysis (the green line) and the b-tagging analysis (cyan line) are shown for 60 fb$^{-1}$. The area above a line is reachable in that channel. [10]

## 8.8 Discovery reach in $\tan\beta - m_A$ plane

The $5\sigma$ discovery significance was calculated with Poisson statistics [88] of the number of events within each mass window and the background uncertainties from the previous section. The reach for the three $m_A$ scenarios can be seen in Table 8.15 and Figure 8.16 along with the values for the b-tagging study.

| Analysis | | $m_A = 200\,\mathrm{GeV/c^2}$ | $m_A = 500\,\mathrm{GeV/c^2}$ | $M_A = 800\,\mathrm{GeV/c^2}$ |
|---|---|---|---|---|
| $\mathrm{E_T^{miss}}$ | no systematics | 19 | 27 | 41 |
| | with systematics | 22 | 31 | 48 |
| b-tagging | no systematics | 20 | 32 | 46 |
| | with systematics | 21 | 34 | 49 |

Table 8.15: Summary of expected events for all three mass scenarios for 60fb$^{-1}$ in the $m_h^{max}$ scenario.

Replacing the b-tagging event selection with an $\mathrm{E_T^{miss}}$ cut resulted in improved $\tan\beta$ sensitivity for $m_A = 500$ and $800\,\mathrm{GeV/c^2}$. However at low values ($m_A \approx 200\,\mathrm{GeV/c^2}$) the amount of $\mathrm{E_T^{miss}}$ in signal and background events was similar. This resulted in a cut that removed a significant amount of the signal ($\sim 90\%$). At the highest values of $m_A \approx 800\,\mathrm{GeV/c^2}$ the low number of signal events and the large background uncertainty reduced the sensitivity.

## 8.9 Conclusion

It was concluded that CMS was sensitive to the MSSM heavy Higgs boson in the range $200 < m_A < 800\,\mathrm{GeV/c^2}$ with $\tan\beta \geq 48$. This was slightly better than the previous b-tagging study except at $m_A = 200\,\mathrm{GeV/c^2}$.

When this analysis is performed on real data it will be combined with the b-tagging study to obtain the greatest reach. The other channels shown in Figure 8.16 will also be combined with the aim of providing the greatest possible reach. These will be combined with general MSSM searches to perform a fit of the MSSM parameter space.

Any improvement in the $\mathrm{E_T^{miss}}$ performance of the CMS experiment would benefit this channel. It is likely that both the $\mathrm{E_T^{miss}}$ scale and resolution will be improved when CMS implements energy flow. This would increase reach in the $\tan\beta - m_A$ plane especially for $m_A \approx 200\,\mathrm{GeV/c^2}$.

# Chapter 9

# Conclusion

One of the most important CMS grid activities is that of distributed user analysis. GROSS was created as a prototype distributed analysis tool. This tool performed well, submitting almost 4,000 jobs with a success rate of almost 90%. Nevertheless it was decided to discontinue development in favour of a similar tool.

The experience gained from GROSS development and usage was used in the redesign of one of the core pieces of CMS software, BOSS. This software was designed to be used within the CMS production system and was adopted by GROSS for use with analysis. BOSS is currently used by both production and analysis systems within CMS and runs tens of thousands of jobs per week.

The $\tau$ energy scale has been investigated and Monte Carlo corrections produced for use by the CMS physics community. It is envisaged that this correction will be maintained until CMS particle flow algorithms become sufficiently sensitive. During experiment startup when the particle flow is still being investigated it may be possible to use QCD jet + photon events along with $\tau$ tagging to calibrate the energy scale. Investigation has shown that this may be a possibility but that the $\tau$ tagging criteria will have to be improved.

The study in this thesis has shown that both the $A$ and $H$, if they exist, may be seen at CMS in the channel $gg \rightarrow A/H \rightarrow \tau\tau \rightarrow$ two jets with 60fb$^{-1}$ of data. This study has been performed using an $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ selection instead of, as studied previously, b-tagging. It has been seen that this channel provides a $5\sigma$ discovery significance for $m_A = 200\,\mathrm{GeV/c}^2$ with $\tan\beta \geq 22$, $m_A = 500\,\mathrm{GeV/c}^2$ with $\tan\beta \geq 31$ and $m_A = 800\,\mathrm{GeV/c}^2$ with $\tan\beta \geq 48$. This is better than the previous study except at $m_A = 200\,\mathrm{GeV/c}^2$.

It is envisaged that, once running, CMS will implement an energy flow algorithm which will improve its $\mathrm{E}_\mathrm{T}^\mathrm{miss}$ scale and resolution. This should improve the performance of this study at low $m_A$. This study will be combined with the b-tagging and other studies to obtain the greatest reach in the $\tan\beta - m_A$ plane.

# Appendix A

# Acronyms

**ALICE** A Large Ion Collider Experiment.

**AliEn** Alice EnviroNment.

**AOD** Analysis Object Data.

**APD** Avalanche PhotoDiode.

**API** Application Programming Interface.

**ATLAS** A Toroidal LHC Apparatus.

**BOSS** Batch Object Submission System.

**BR** Branching Ratio.

**CMS CAF** CMS Cern Analysis Facility.

**CE** Computing Element.

**CLI** Command Line Interface.

**CMS** Compact Muon Solenoid.

**CPU** Central Processing Unit.

**CRAB** Cms Remote Analysis Builder.

**CSC** Cathode Strip Chambers.

**DB** Database.

**DBS** Dataset Bookeeping System.

**DIRAC** Distributed Infrastructure with Remote Agent Control.

**DLS** Data Location Service.

**DM** Data Management

**DOM** DOcumnet Model.

**DQM** Data Quality Monitoring.

**DT** Drift Tube.

**EB** Ecal Barrel.

**ECAL** Electromagnetic CALorimeter.

**EDG** European DataGrid.

**EE** Electromagnetic calorimter Endcap.

**EGEE** Enabling Grids for E-sciencE.

**FTP** File Transfer Protocol.

**GCT** Global Calorimeter Trigger.

**GROSS** GRidified Orca Submision System.

**GUID** Globally Unique IDentifier.

**HB** Hadron calorimeter Barrel.

**HCAL** Hadron CALorimeter.

**HE** Hadron Endcap.

**HEPCAL** HEP common application layer for analysis

**HF** Hadron calorimeter Forward.

**HI** Heavy Ion.

**HLT** High Level Trigger.

**HO** Hadron calorimter Outer.

**HTML** HyperText Markup Language.

**II** Information Index.

**IO** Input Output.

**JDL** Job Description Language.

**JIM** Job and Information Monitoring

**LB** Logging and Bookeeping.

**LCG** Lhc Computing Grid.

**LFN** Logical File Name.

**LHC** Large Hadron Collider.

**LRC** Local Replica Catalog.

**MC** Monte Carlo.

**MSSM** Minimal Supersymmetric Standard Model.

**ORCA** Object-oriented Reconstruction for CMS Analysis.

**OSG** Open Science Grid.

**PFN** Physical File Name.

**PRS** Physics Reconstruction and Selection.

**RB** Resource Broker.

**RLI** Replica Location Index.

**RLS** Replica Location Service.

**RMC** Replica Management Catalogue.

**RPC** REsistive Plate Chamber.

**SAM** Sequential Access via Metadata.

**SAX** Simple Api for Xml.

**SE** Storage Element.

**SM** Standard Model.

**SOAP** Simple Object Access Protocol.

**SPS** Super Proton Synchrotron.

**SQL** Structured Query Language.

**SRM** Storage Resource Manager.

**SUSY** SUperSYmmetry.

**SWIG** Simplified Wrapper and Interface Generator.

**TB** Test Beam.

**TDR** Technical Design Report.

**TEC** Tracker EndCap.

**TIB** Tracker Inner Barrel.

**TID** Tracker Inner Disk.

**TOB** TRacker Outer Barrel.

**UI** User Interface.

**VO** Virtual Organisation.

**WM** Workflow Management.

**WMS** Workload Management System.

**WN** Worker Node.

**XML** eXtensible Markup Language.

**XSL** Xml Stylesheet Language.

# References

[1] H. Tallini, D. Colling, B. Macevoy, and S. Wakefield, "GROSS: an end user tool for carrying out batch analysis of CMS data on the LCG-2 Grid.", in *Proceedings of CHEP04, Interlaken, Switzerland.* 2004.

[2] W. Bacchi, G. Codiispoti, C. Grandi, D. Colling, B. Macevoy, S. Wakefield, and Y. Zhang, "Evolution of BOSS, A tool for job submission and tracking", in *Proceedings of CHEP06, Mumbai, India.* 2006.

[3] P. W. Higgs, "Broken symmetries, massless particles and gauge fields", *Physics Letters* **12** 132–133, (September, 1964). doi:10.1016/0031-9163(64)91136-9.

[4] S. P. Martin, "A Supersymmetry Primer", hep-ph/9709356.

[5] "The Large Hadron Collider: Conceptual Design", *CERN/AC/95-05 (LHC)* (1995).

[6] CMS Collaboration, "The Compact Muon Solenoid Letter of Intent", *CERN/LHCC* **LHCC/I 1** (1992).

[7] CMS Collaboration, "The Compact Muon Solenoid Technical Proposal", *CERN/LHCC* **94-38** (1994).

[8] "First proton-antiproton collisions in the CERN SPS collider", *Physics Letters B* **107** 306–309, (December, 1981). doi:10.1016/0370-2693(81)90836-4.

[9] CMS Collaboration, "The CMS Physics Technical Design Report, Volume 1", *CERN/LHCC* **2006-001** (2006).

[10] CMS Collaboration, "The CMS Physics Technical Design Report, Volume 2", *CERN/LHCC* **2006/021** (2006).

[11] CMS Collaboration, "The Tracker Project Technical Design Report", *CERN/LHCC* **98-006** (1998).

[12] CMS Collaboration, "The Electromagnetic Calorimeter Technical Design Report", *CERN/LHCC* **97-033** (1997).

[13] CMS Collaboration, "The Hadron Calorimeter Technical Design Report", *CERN/LHCC* **97-031** (1997).

[14] CMS Collaboration, "The Muon Project Technical Design Report", *CERN/LHCC* **97-32** (1997).

[15] CMS Collaboration, "The Magnet Project Technical Design Report", *CERN/LHCC* **97-010** (1997).

[16] C. Seez, "The CMS trigger system", *The European Physical Journal C.* **34** s1.151–s1.159, (2004). doi:10.1140/epjcd/s2004-04-016-8.

[17] "SPECINT2000 benchmark." http://www.spec.org/.

[18] C. Kesselman and I. Foster, "The Grid: Blueprint for a New Computing Infrastructure". Morgan Kaufmann Publishers, November, 1998.

[19] M. Greenberger, "Computers and the world of the future", *The Atlantic Monthly* (1962).

[20] "European DataGrid Project (EDG)." http://eu-datagrid.web.cern.ch/eu-datagrid/.

[21] "Enabling Grids for E-sciencE (EGEE)." http://public.eu-egee.org/.

[22] "LHC Computing Grid Technical Design Report", *CERN/LHCC* **2005-024** (2005).

[23] "The Virtual Data Toolkit." http://vdt.cs.wisc.edu.

[24] "International Virtual Data Grid Laboratory." http://www.ivdgl.org.

[25] "Grid Physics Network." http://www.griphyn.org.

[26] http://datatag.web.cern.ch/datatag.

[27] P. J. W. Faulkner et al., "GridPP: development of the UK computing Grid for particle physics", *Journal of Physics G: Nuclear and Particle Physics* **32** N1–N20, (January, 2006). doi:10.1088/0954-3899/32/1/N01.

[28] M. Aderholz, "Models of Networked Analysis at Regional Centres for LHC Experiments (MONARC) - Phase 2 Report", *CERN/LCB* **2000-001** (2000).

[29] C. Anglano et al., "Integrating GRID Tools to Build a Computing Resource Broker: Activities of DataGrid WP1", in *Proceedings of CHEP01, Beijing, China*. 2001.

[30] "LCG-2 User Guide." https://edms. cern. ch/file/454439//LCG-2-UserGuide.pdf.

[31] M. Solomon, "The ClassAd Language Reference Manual." http://www.cs.wisc.edu/condor/classad.

[32] "LHC Persistency framework, Pool Of persistent Objects for LHC." http://pool.cern.ch/.

[33] "MySQL website." http://www.mysql.com.

[34] "CMS Computing Model : The CMS Computing Model RTAG"', *CMS NOTE* **2004-031** (December, 2004).

[35] "Revised CMS computing requirements." http://lcg.web.cern.ch/LCG/MB.

[36] R. Brun and F. Rademakers, "ROOT - An Object Oriented Data Analysis Framework", in *AIHENP'96 Workshop, Lausane*, volume 389, pp. 81–86. 1996.

[37] "The CDF detector: an overview", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **271** 387–403, (September, 1988). doi:10.1016/0168-9002(88)90298-7.

[38] "The D0 detector", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **338** 185–253, (January, 1994). doi:10.1016/0168-9002(94)91312-9.

[39] "Run II Computing Review." http://cdinternal.fnal.gov/RUNIIRev/runIIMP05.asp, 2005.

[40] M. Corvo, "CRAB: a tool to enable CMS Distributed Analysis", in *Proceedings of CHEP06, Mumbai, India.* 2006.

[41] K. Harrison et al., "GANGA: a user-Grid interface for Atlas and LHCb", in *Proceedings of CHEP03, La Jolla, California.* 2003. cs.se/0306085.

[42] J. Rehn, "PhEDEx high-throughput data transfer management system", in *Proceedings of CHEP06, Mumbai, India.* 2006.

[43] M. Branco, "Don Quijote - Data Management for the ATLAS Automatic Production System", in *Proceedings of CHEP04, Interlaken, Switzerland.*

[44] P. Saiz, L. Aphecetche, P. Buncic, R. Piskac, and V. Sego, "AliEn–ALICE environment on the GRID", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **502** 437–440, (April, 2003). doi:10.1016/S0168-9002(03)00462-5.

[45] N. Brook et al., "DIRAC - Distributed Infrastructure with Remote Agent Control", in *Proceedings of CHEP03, La Jolla, California.* 2003. cs.dc/0306060.

[46] C. Grandi, "CMS distributed data analysis challenges", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **534** 87–93, (November, 2004). doi:10.1016/j.nima.2004.07.065.

[47] S. Wynhoff, "Using the reconstruction software, ORCA, in the CMS datachallenge 2004", in *Proceedings of CHEP04, Interlaken, Switzerland.* 2004.

[48] S. Agostinelli, "G4 - a simulation toolkit", *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **506** 250–303, no. 3,.

[49] J. Andreeva et al., "Use of grid tools to support CMS distributed analysis", in *Nuclear Science Symposium Conference Record*, volume 4, pp. 2029–2032 Vol. 4. 2004.

[50] A. Fanfani, "Distributed computing grid experiences in CMS data challenge", in *Proceedings of CHEP04, Interlaken, Switzerland.*

[51] C. Grandi, "BOSS: a tool for batch job monitoring and book-keeping", in *Proceedings of CHEP03, La Jolla, California.* May, 2003. hep-ex/0305098.

[52] G. E. Graham, D. Evans, and I. Bertram, "McRunjob: A High Energy Physics Workflow Planner for Grid Production Processing", in *Proceedings of CHEP03, La Jolla, California.* Jun, 2003. cs.DC/0305063.

[53] "LHC Grid Computing Project: COMMON USE CASES FOR A HEP COMMON APLLICATION LAYER FOR ANALYSIS (HEPCAL)", *LHC-SC2-20-2002* (October, 2003).

[54] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Design Patterns". Addison-Wesley Professional, January, 1995.

[55] H. Tallini and S. Wakefield, "GROSS 0.4.0 User and Installation Guide." http://www.hep.ph.ic.ac.uk/e-science/projects/downloads.html.

[56] J. Generowicz et al., "SEAL: Common Core Libraries and Services for LHC Applications", in *Proceedings of CHEP03, La Jolla, California.* 2003. physics/0306033.

[57] D. Abrahams et al., "BOOST libraries for C++." http://www.boost.org.

[58] J. Andreeva et al., "Distributed Computing Grid Experiences in CMS", *IEEE Transactions on Nuclear Science* **52** 884–890, (2005), no. 4,.

[59] D. Cameron and E. Al, "Replica Management in the European DataGrid Project", *Journal of Grid Computing* **2** 341–351, no. 4,. doi:10.1007/s10723-004-5745-x.

[60] M. Girone et al., "Experience with POOL in the LCG data challenges of three LHC experiments", in *Proceedings of CHEP04, Interlaken, Switzerland.* 2004.

[61] J. P. Baud and J. Casey, "Evolution of LCG-2 Data Management", in *Proceedings of CHEP04, Interlaken, Switzerland, September.* 2004.

[62] "Scientific Linux CERN 3." http://linux.web.cern.ch/linux/scientific3.

[63] "RedHat Enterprise Linux." http://www.redhat.com/rhel.

[64] D. Bonacorsi et al., "Running CMS software on GRID Testbeds", in *Proceedings of CHEP03, La Jolla, California.* 2003. physics/0306038.

[65] H. B. Newman, I. C. Legrand, P. Galvez, R. Voicu, and C. Cirstoiu, "MonALISA : A Distributed Monitoring Service Architecture", in *Proceedings of CHEP03, La Jolla, California.* Jun, 2003. cs.DC/0306096.

[66] D. M. Beazley, "Automated scientific software scripting with SWIG", *Future Gener. Comput. Syst.* **19** 599–609, (July, 2003). doi:10.1016/S0167-739X(02)00171-1.

[67] "SIP website." http://www.riverbankcomputing.co.uk/sip/.

[68] D. Abrahams and R. W. Grosse-Kunstleve, "Building Hybrid Systems with Boost. Python", *C/C++ Users Journal* **21** (July, 2003).

[69] R. Geus and O. a. Skavhaug, "Python Wrapper Tools; a Performance Study", in *Proceedings of EuroPython 2004, Gothenburg, Sweden.*

[70] F. Halzen and A. D. Martin, "Quarks and Leptons." Wiley, 1985.

[71] D. Griffiths, "Introduction to elementary particles." Wiley, 1987.

[72] A. Salam and J. C. Ward, "Electromagnetic and weak interactions", *Phys. Lett.* **13** 168–171, (1964).

[73] D. A. Dicus and V. S. Mathur, "Upper Bounds on the Values of Masses in Unified Gauge Theories", *Physical Review D* **7** 3111+, (May, 1973). doi:10.1103/PhysRevD.7.3111.

[74] G. Kane, C. Kolda, and J. D. Wells *Phys. Rev. Lett. J. C.* **25** 113, (2002).

[75] W. M. Yao et al., "Review of Particle Physics", *Journal of Physics G* **33** 1+, (2006).

[76] G. Bagliesi et al., “Tau jet reconstruction and tagging with CMS”, *Eur. Phys. J. C.* 1–21, (2006). doi:10.1140/epjcd/s2006-02-001-y.

[77] S. Schael et al., “Search for neutral MSSM Higgs bosons at LEP”, *Eur. Phys. J.* **C47** 547–587, (2006), hep-ex/0602042.

[78] M. Carena, S. Heinemeyer, C. E. M. Wagner, and G. Weiglein, “MSSM Higgs Boson Searches at the Tevatron and the LHC: Impact of Different Benchmark Scenarios”, hep-ph/0511023.

[79] M. Carena and H. E. Haber, “Higgs Boson Theory and Phenomenology”, hep-ph/0208209.

[80] S. Gennai, A. Nikitenko, and L. Wendland, “Search for MSSM heavy netral Higgs boson in the $\tau\tau \rightarrow$ two jet decay mode”, *CMS NOTE* **2006-126**.

[81] T. Sjöstrand, P. Edeacuten, C. Friberg, L. Lönnblad, G. Miu, S. Mrenna, and E. Norrbin, “High-Energy-Physics Event Generation with PYTHIA 6.1”, hep-ph/0010017.

[82] S. Jadach, Z. Ws, R. Decker, and J. H. Kühn, “The $\tau$ decay library TAUOLA, version 2.4”, *Computer Physics Communications* **76** 361–380, (August, 1993). doi:10.1016/0010-4655(93)90061-G.

[83] A. Heister et al., “Jet reconstruction and performance in the CMS detector”, *CMS NOTE* **2006-036** (2006).

[84] A. Nikitenko, S. Kunori, and R. Kinnunen, “Missing Transverse Energy Measurment with Jet Energy Correction”, *CMS NOTE* **2001/040** (2001).

[85] S. Heinemeyer, W. Hollik, and G. Weiglein, “FeynHiggs: a program for the calculation of the masses of the neutral CP-even Higgs bosons in the MSSM”, hep-ph/9812320.

[86] J. Campbell, R. K. Ellis, and D. Rainwater, “Next-to-leading order QCD predictions for W+2j and Z+2j production at the CERN LHC”, hep-ph/0308195.

[87] A. Kalinowski and A. Nikitenko, “Measurement of the $\tau$ tag efficiency using the $Z \rightarrow \tau\tau \rightarrow \mu + hadrons + X$ events”, *CMS NOTE* **2006/074** (2006).

[88] S. I. Bityukov, S. E. Erofeeva, N. V. Krasnikov, and A. Nikitenko, “Program for evaluation of significance, confidence intervals and limits by direct calcualtion of probabilities.”, in *Proceedings of PhyStat05, Oxford, UK.* 2005.